jamk.fi

Jyvälä.fi - Verkkosivuston toteutus

Vesa-Tapani Vertainen

Opinnäytetyö

Toukokuu 2018

Tekniikan ja liikenteen ala

Insinööri (AMK), Ohjelmistotekniikan koulutusohjelma

Jyväskylän ammattikorkeakoulu

JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Vertainen, Vesa-Tapani	Opinnäytetyö, AMK	toukokuu 2018
	Sivumäärä 53	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi
Jyvälä.fi – Verkkosivuston toteutus

Tutkinto-ohjelma
Ohjelmistotekniikan koulutusohjelma

Työn ohjaaja(t)
Rantala Ari

Toimeksiantaja(t)
Jyvälän Setlementti ry

Tiivistelmä

Opinnäytetyön tavoite oli toteuttaa Jyvälän Setlementti ry:lle uudet verkkosivut. Tärkeää oli saada sivusto selkeäksi vierailijan kannalta ja toisaalta mahdollisimman helpoksi sivuston päivittäjille. Vanhan sisällönhallintajärjestelmän ei nähty olevan paras mahdollinen käyttötarkoitusta ajatellen. Lisäksi ulkoasu haluttiin uudistaa Suomen setlementtiliiton uusien graafisen ilmeen ohjeiden mukaiseksi. Myös Jyvälän kansalaisopiston ilmoittautumista haluttiin selkeyttää niin, ettei kursseja tarvitse selata kahdesta eri järjestelmästä.

Aluksi tutkittiin eri tapoja uuden, selkeämmän verkkosivuston toteuttamiseksi. Vaihtoehtoina oli ohjelmoida sivut kokonaan itse, käyttää sovelluskehyksiä tai käyttää alustana jotakin julkaisujärjestelmää. Sivuston helpon päivitettävyyden vaatimuksen takia omasta ohjelmoinnista luovuttiin. Jyvälän Setlementin tapauksessa sovelluskehyksien käytöllä ei myöskään nähty olevan etua julkaisujärjestelmään verrattuna. Yleisimpiä julkaisujärjestelmiä vertailtiin asentamalla ne virtuaalipalvelimeen ja testaamalla niiden ominaisuuksia. Julkaisujärjestelmäksi valikoitui WordPress. Merkittävin tekijä valinnassa oli, että WordPressin pohjalle rakennettu sivu on helpoin hallita ja päivittää myös sellaisen henkilön, jolla ei ole aiempaa kokemusta julkaisujärjestelmistä. Myös saatavilla olevien ohjeiden ja helposti käyttöönotettavien lisäosien määrä oli osatekijä. Verkkosivusto toteutettiin lapsiteemana Karuna-teeman pohjalta. Lapsiteemalla saatiin käyttöön kaikki Karuna-teeman ominaisuudet, mutta haluttuihin osioihin voitiin toteuttaa muutoksia ja lisätoiminnallisuuksia, kuten kansalaisopiston kurssivalikko. Osa toiminnoista toteutettiin omaan pluginiin.

Verkkosivuista saatiin WordPress-julkaisujärjestelmällä hyvin toimivat, kävijän kannalta selkeät sekä ylläpidolle helpot päivittää. Myös kurssi-ilmoittautuminen saatiin suoraviivaisemmaksi.

Avainsanat (asiasanat)
Web-kehitys, julkaisujärjestelmä, Wordpress, PHP, MySQL

Muut tiedot

jamk.fi

Description

Author(s)	Type of publication	Date
Vertainen, Vesa-Tapani	Bachelor's thesis	May 2018
		Language of publication: Finnish
	Number of pages 53	Permission for web publication: x

Title of publication
Jyvälä.fi – web site implementation

Degree programme
Software Engineering

Supervisor(s)
Rantala Ari

Assigned by
Jyvälän Setlementti ry

Abstract

The objective of the thesis was to implement a new web site for Jyvälän Setlementti ry. The essential issues were that the site should deliver the users a great experience with its clear appearance and on the other hand, the site should be easy to maintain and update. The previously used content management system, CMS, was not considered to be the best option for the purpose. In addition, the layout of the site was to be renewed to match the graphical layout guidelines of the Finnish Federation of Settlement Houses. The course registration system needed to be enhanced as well.

At first, different ways to implement a new, clearer web site were examined. The options were to code the site from scratch, use frameworks or a content management system. For the need of the site to be easily updated, coding from scratch was discarded. In this case, the use of frameworks would not have been a notable advantage in comparison to a CMS. The most common CMSs were installed on a virtual machine where their features were tested and compared. As a result, WordPress was chosen to be used as the CMS. The most remarkable factor in selecting the CMS was, that a web site built on WordPress would be the easiest to maintain and update, even for a person without preceding experience on a CMS. The large amount of instructions and plugins available for WordPress was also a significant advantage. The implementation of the web site was carried out as a child theme based on the Karuna theme. As a child theme, all features of Karuna were deployed; however, changes and additional features were applied where needed, e.g. in the course selection menu. Some features were implemented as a plugin.

The result was a well working site that is easy to maintain and provides a good user-experience. The course registration system was also streamlined.

Keywords/tags (subjects HYPERLINK "http://vesa.lib.helsinki.fi/")

Web-development, content management system, Wordpress, PHP, MySQL

Miscellaneous

Sisältö

Kuviot

Taulukot

Sanasto

Alusta

Alusta, jossa ohjelmistoa käytetään, tarkoittaa sekä laitetta että käyttöjärjestelmää.

Järjestelmävaatimukset

Järjestelmävaatimus kertoo, minkälainen järjestelmän on oltava tai kuinka sen on suoriuduttava.

Kehitysympäristö (ohjelmointiympäristö, IDE)

Kehitysympäristö tarkoittaa ohjelmointiin käytettävää ohjelmistoa.

Mobiili

Mobiili tarkoittaa helposti liikuteltavaa, tietoliikenteessä yleensä langatonta laitetta.

Ohjelmointirajapinta (API)

Ohjelmointirajapinta on käyttöjärjestelmän ja sovelluksen (tai jonkun muun kahden ohjelman) välinen rajapinta, joka yksinkertaistaa ohjelmointia.

Sisällönhallintajärjestelmä

(ts. julkaisujärjestelmä, engl. Content Management System, CMS). Sovelluskokonaisuus, jota käyttäen verkkosivujen julkaisu on nopeaa ja helppoa.

Verkkosivu

Internetissä olevaan merkintäkieliseen tiedostoon perustuva tietokokonaisuus, joka voidaan esittää käyttäjän laitteistolla. Verkkosivusto on tiettyä aihetta käsittelevä verkkosivujen joukko.

1. Työn lähtökohdat

1.1 Tausta

Työn aiheena oli Jyvälän Setlementti ry:n verkkosivuston modernisointi. Syy toimeksiantoon oli vanhentunut sivusto, joka oli ajan kuluessa laajentunut varsin sekavaksi. Vaatimuksena oli, että tehtävään valittu henkilö voi toteuttaa sivuston itsenäisesti, pääosin etätyönä, mutta rakenteen, visuaalisen ilmeen ja sisällön suunnittelu tuli tehdä yhteistyössä Jyvälän Setlementin viestintätiimin kanssa. Työn haluttiin käynnistyvän syksyllä 2017 tai viimeistään vuoden 2018 alussa.

1.2 Jyvälän Setlementti ry

1.2.1 Esittely

Setlementtiliike syntyi Englannissa 1800-luvun lopussa, kun Lontoon köyhille alueille perustettiin keskuksia kansalaisten tueksi yliopisto-opiskelijoiden toimesta. Ensimmäinen setlementti sai alkunsa 1884. Suomeen liike tuli varsin pian tämän jälkeen, kun Helsingin Kallioon perustettiin Kansankoti jo 1890-luvun lopulla. Sieltä liike levisi ympäri Suomea. Lastentarhat, nuorisotyö sekä luento- ja keskustelutilaisuudet, joilla pyrittiin yhdistämään eri kansalaisryhmiä, olivat ensimmäisiä työmuotoja. (Historia n.d.)

Jyvälän Setlementti on kansalaisjärjestö, jonka perusta on setlementtiliikkeessä, ja se kuuluu Suomen setlementtiliittoon. Toiminnan tarkoitus on muun muassa tukea kansalaisten henkistä kasvua, elämänhallintaa ja edistää eettisyyttä. Toiminta keskittyy yhteiskunnan tarpeisiin. (Jyvälän Setlementti n.d.)

1.2.2 Jyvälän monet toiminnot

Jyvälän keskeisiä toimintoja ovat kansalaisopiston kurssit liittyen mm. kehon ja mielen hyvinvointiin, koululaisten iltapäivätoiminta Jyvälän Jälkkärit, nuorisotoiminta harrastusten, leirien ja tapahtumien muodossa, kansalaistoiminta, johon kuuluu

muun muassa kohtaamispaikan tarjoaminen lähialueen eri ikäryhmiin kuuluville, sekä hanketoiminta ajankohtaisia haasteita varten. (Jyvälän Setlementti n.d.)

Jyvälän toimintoihin kuuluu myös sukupuolisensitiivinen erityisnuorisotyö, Oiva ja Ilona, jossa nuorten kasvua ja elämänhallintaa pyritään tukemaan yksilötapaamisilla, teemaryhmillä ja muulla toiminnalla. Tähän toimintaan kuuluu myös aihepiirin asiantuntijana toimiminen. (Jyvälän Setlementti n.d.) Oivan tarkoitus on toimia kohtaamispaikkana pojille ja nuorille miehille, Ilonan puolestaan tytöille ja nuorille naisille (Oiva ja Ilona n.d).

1.2.3 Jyvälän sivustot

Verkkosivut on tällä hetkellä jaettu niin, että Jyvälän varsinainen verkkosivusto on osoitteessa www.jyvala.fi, ja Oiva ja Ilona -työllä on oma verkko-osoite www.oiva-ilona.fi. Oiva ja Ilona -sivun käyttöliittymä sekä graafinen ilme eroavat merkittävästi Jyvälän sivustosta. Jyvälän sivut on jaoteltu eri osa-alueisiin, muun muassa toimintoihin, hankkeisiin ja projekteihin sekä vapaaehtoistehtäviin, jotka puolestaan sisältävät useita muita osioita. Yksi tärkeimmistä osioista ovat kansalaisopiston kurssit ja niiden ilmoittautumistoiminto. Lisäksi käytössä on useita alidomaineja, kuten kansalaisopisto.jyvala.fi, oi.jyvala.fi ja avaimet.jyvala.fi.

1.2.4 Kansalaisopiston kurssinhallintajärjestelmä

Kansalaisopiston kursseille ilmoittautuminen tapahtuu ulkopuolisen järjestelmän, Hellewin, kautta. Sivuston suora osoite on www.opistopalvelut.fi/jyvala/. Hellewi on kurssinhallintajärjestelmä, joka kehitettiin jo vuonna 1985 ja myöhemmin siirrettiin internetissä toimivaksi. Hellewin kautta voidaan kurssien hallinnan ja ilmoittautumisjärjestelmän lisäksi hoitaa muun muassa laskutusta, kirjanpitoa ja tilojenhallintaa. (Hellewi n.d.)

1.3 Lähtötilanne

Jyvala.fi -sivuston sisältö oli vaikeasti hahmotettavissa niin sivustolla vierailevan kuin myös sisällöntuottajan näkökulmasta. Palautteen perusteella oli huomattu, että asiakkaiden on vaikea löytää sivuilta haluamiaan tietoja. Toisaalta sisällönhallintajärjestelmästä oli vaikea löytää oikeaa kohtaa, josta tietyn sivun päivittäminen tapahtuu.

1.3.1 Palvelimen ominaisuudet

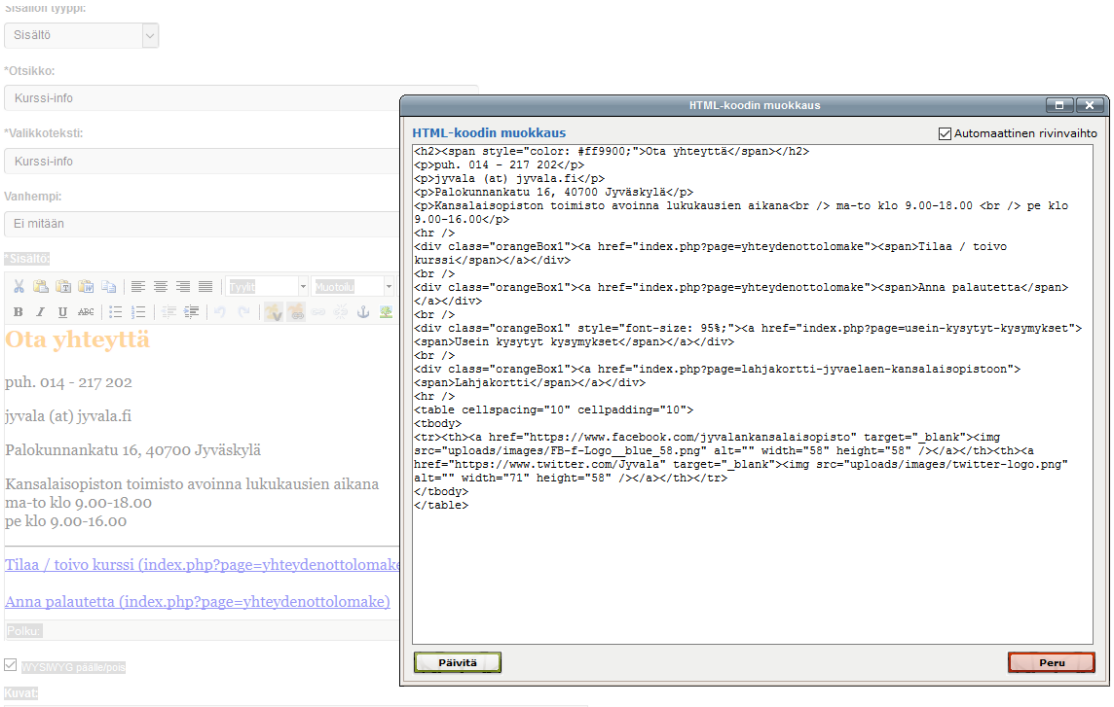
Palveluntarjoajan kautta käytössä ollut palvelintila oli 1,17 gigatavua ja SQL-tietokantoja käytettävissä 10, joista työn alkaessa käytössä oli 8. Palvelimen hallinta tapahtuu graafisen CPanel käyttöliittymän kautta. Suoraan CPanelin kautta on mahdollista asentaa myös sisällönhallintajärjestelmiä, kuten WordPress, Drupal ja Joomla.

1.3.2 Sisällönhallintajärjestelmä

Käytössä ollut sisällönhalllintajärjestelmä oli CMS Made Simple eli CMSMS, joka on julkaistu vuonna 2004. Jyvälän käytössä ollut CMSMS oli versionumeroltaan 1.11.4, joka on julkaistu vuonna 2012 (Announcing CMSMS 1.11.4 Fernandina 2012).

CMS Made Simple on avoimen lähdekoodin järjestelmä, joka perustuu PHP- ja MySQL -tekniikoihin. Oman sivustonsa mukaan CMSMS sopii parhaiten yritysten ja organisaatioiden sivustojen ylläpitoon, ei niinkään blogien tai artikkelipohjaisen sisällön näyttämiseen. CMSMS:n tavoite on olla riittävän laaja ja monipuolinen, mutta samalla yksinkertainen, eikä ominaisuuksissaan yliampuva, kuten monet vastaavat järjestelmät. Sivustoa hallitaan useimpien sisällönhallintajärjestelmien tapaan selaimen kautta. (CMS Made Simple 2.x Official Documentation n.d.)

Content Manager -moduulin avulla päästään muun muassa lisäämään sivuja ja muokkaamaan ulkoasua teemojen kautta (About CMS Made Simple n.d). Kuviossa 1 näkyy Kurssi-info-artikkelin muokkaus CMSMS-järjestelmässä.



Kuvio 1. Kurssi-info -artikkelin muokkaus CMS Made Simple -järjestelmässä

2. Verkkosivustojen rakenne ja sisällönhallinta

2.1 HTML-kieli ja sisällönhallintajärjestelmät

Nykypäivänä verkkosivustot usein rakennetaan jonkin sisällönhallintajärjestelmän ts. julkaisujärjestelmän päälle. Vielä vuosituhannen vaihteessa oli tyypillisempää koodata sivut itse HTML-kielellä tai käyttää WYSIWYG HTML-editoreita kuten FrontPage ja Dreamweaver. Yrityksillä oli myös tapana rakentaa sivunsa kokonaan uudestaan aika ajoin, mutta julkaistavan sisällön kasvaessa ja julkaisujärjestelmien kehittyessä tuli järkevämmäksi vaihtoehdoksi tehokkaampaan sisällönhallintaan siirtyminen (Tolvanen 2007, 5).

Myös tämän päivän tunnetuimpien sisällönhallintajärjestelmien kehitys on aloitettu 2000-luvun alkuvuosina. Käytännössä tämä muutos on tarkoittanut sitä, että kun aikaisemmin sivustot saattoivat pysyä staattisena pitkäänkin, nykyään niiden sisältöä voidaan päivittää päivittäisellä tasolla julkaisujärjestelmien ansiosta. Toisaalta se on myös luonut painetta sivujen päivitystahdin ylläpitämiseen.

Sisällönhallintajärjestelmällä voidaan tarkoittaa eri tyyppisiä ratkaisuja. Se voi olla pelkästään blogien julkaisuun tarkoitettu, tai se voi kattaa lähes kaiken mahdollisen digitaalisen informaation hallinnan, kuten tekstin ja kuvien lisäksi yrityksen intranetin, sähköpostit, laskutuksen ja lomakkeet. Se voi olla myös verkkokauppasovellus. Tyypillisesti sisällönhallintajärjestelmää käyttäessä käyttäjän ei ole välttämätöntä tietää mitään verkkosivujen taustalla olevasta HTML-kielestä. (Pouru 2012, 7-8.)

Tämä ohjelmointitaidon tarpeettomuus on varmasti edesauttanut suuresti useiden sivustojen, erityisesti blogien, syntymistä. Varsinkin Wordpressin suosio blogisivujen alustana on lähtöisin sen äärettömän helposta käyttöliittymästä. Myöhemmin Wordpress on laajentunut hyvin monipuoliseksi sisällönhallintajärjestelmäksi. Helpoista sisällönhallintajärjestelmistä huolimatta verkkosivuja on edelleen mahdollista tehdä ja myös tehdään HTML-kielellä ohjelmoimalla. Vuosien varrella HTML-kieli on kehittynyt tehokkaammaksi. Myös sivujen ulkoasun muotoiluun käytettyjen tyylitiedostojen eli CSS:n, kehittyminen sekä erilaiset kirjastot, kuten jQuery, ovat helpottaneet ja monipuolistaneet sivujen ohjelmointia.

Sisällönhallintaa käyttäessä ei tarvita välttämättä ohjelmointitaitoja, joten se on tapa, jolla periaatteessa kuka tahansa voi aloittaa sivustojen rakentamisen. Toisaalta myös toisessa ääripäässä, suurissa yrityssivustoissa, voidaan tarvita sisällönhallintajärjestelmiä, koska muuten niiden hallinta olisi haastavaa, ellei jopa mahdotonta. Pelkästään omalla koodilla tehtyjä sivuja käytetäänkin yleensä vain pienemmissä projekteissa, mutta nykypäivänä erittäin tyypillinen tapa tehdä isot verkkosivustot on käyttää pohjana julkaisujärjestelmää ja koodata sen sisään tarvittavat spesiaalit osiot itse.

Etuina sisällönhallintajärjestelmässä ovat myös siihen sisältyvä verkkosivun skaalautuvuus ja dynaamisuus ja muun muassa mahdolliset ajastetut julkaisutoiminnot, versionhallinta ja automatisoitu varmuuskopiointi (Pouru 2012, 11-18). Sisällönhallintajärjestelmien ominaisuuksia voi myös laajentaa lisäosien avulla. Haittapuolena voitaneen pitää sitä, että lähes sata prosenttia julkaisujärjestelmistä vaatii palvelimelta tietokantapalvelun toimiakseen. Muut vaatimukset vaihtelevat järjestelmästä riippuen. Tässä työssä käsiteltyjen järjestelmien vaatimuksia on nähtävissä taulukossa 1 sivulla 15.

2.2 Sovelluskehykset

Valmiin julkaisujärjestelmän ja puhtaan ohjelmoinnin lisäksi vaihtoehtona on myös sovelluskehysten käyttö. Sovelluskehysten ideana on tarjota kehittäjälle valmiita komponentteja ja toteutustapoja verkkopalvelun pohjaksi. Kehys ei ole samalla tavalla valmis paketti kuin sisällönhallintajärjestelmä, vaan sisältää työkaluja esimerkiksi HTML-sivujen ja tietokantatoimintojen tekemiseen. Sovelluskehykset pohjautuvat eri ohjelmointikieliin. Yleisimmät kehykset pohjautuvat PHP:hen, Javaan, Pythoniin, Rubyyn tai Microsoftin .NET-tekniikoihin. (Niemi 2015). Sovelluskehyksiinkin on kuitenkin mahdollista liittää CMS-tyyppinen sisällönhallinta (Kallio 2013).

Sovelluskehyksen käyttö helpottaa suuresti kokonaisuuksien rakentamista, jos verrataan siihen, että kaikki koodattaisiin itse. Eli se sijoittuu edellä mainittujen tekniikoiden välimaastoon. Niemen (2015) mukaan sisällönhallintajärjestelmää kannattanee suosia, jos siitä löytyvät valmiina halutut toiminnot niin, että suurempaa rakenteiden muokkaamista ei tarvita. Sovelluskehyksellä voidaan kuitenkin päästä kevyempiin ratkaisuihin, koska sisällönhallintajärjestelmien mukana saattaa tulla rakenteita tai riippuvuuksia, jotka eivät olisi kyseisen verkkopalvelun kannalta välttämättömiä. Verkkopalvelu voi olla myös niin omanlaisensa, ettei valmiilla sisällönhallintajärjestelmällä sitä pystytä tekemään. Niemi (2015) luettelee sovelluskehyksen käytön eduiksi myös mm. ripeän kehittämisen valmiilla kirjastoilla ja sovelluspaketeilla, tietoturvan ja itsestään dokumentoituvan lähdekoodin. Sovelluskehys siis nopeuttaa ja selkeyttää verkkopalvelujen rakentamista monella tapaa, mutta sen kustannukset voivat kuitenkin olla huomattavastikin suuremmat sisällönhallintajärjestelmän asentamiseen verrattuna, joka on nopein tapa pystyttää verkkosivusto.

3. Työn vaatimusmäärittely

3.1 Toimeksiantajan asettamat tavoitteet

Jyvälän verkkosivuista oli vuosien saatossa tullut monien muutoksien myötä sekavan näköiset. Vaihtelevat nimeämiskäytännöt ja suureksi paisunut tiedostomäärä oli tehnyt myös niiden päivittämisestä hankalaa. Tietyn sivuston kohdan etsiminen päivittämistä varten saattoi viedä paljon aikaa. Yhtenä esimerkkinä epäloogisesta rakenteesta on sivuston oikeassa laidassa oleva yhteydenotto- ja palautepalkki, joka löytyy sivustonhallinnasta nimellä Kurssi-info.

Ensinnäkin työn tarkoitus oli selvittää, millä tavalla sivusto saataisiin näyttämään visuaalisesti paremmalta ja selkeämmältä, miten asiakas löytäisi paremmin haluamansa tiedot ja miten sisältö saataisiin mahdollisimman helposti hallittavaksi. Tutkimustyön jälkeen oli toteutettava sivustolle parhaaksi katsotut muutokset. Tiivistetysti sanottuna tarkoitus oli saada sivujen ulkonäkö nykyaikaiseksi ja selkeäksi sekä rakenteeltaan sellaiseksi, että sisältöä voisi "mattimeikäläinenkin" päivittää. Myös mobiilikäytettävyys tuli ottaa huomioon. Graafiset päälinjaukset, värimaailma ja sisältö tulivat Jyvälän puolesta.

Ulkoasun ja rakenteen lisäksi oli selvitettävä, onko tähänkin asti käytössä olleeseen Hellewi-järjestelmään olemassa rajapintaa tai muuta tapaa, jolla kurssi-ilmoittautumiset voisi hoitaa suoraan sivulta. Jyvälän sivustolla kunkin kurssin esittelyn alta löytyi Ilmoittaudu kurssille -linkki, joka vei Hellewi-järjestelmän Jyvälä-osioon, mutta ei kuitenkaan suoraan kyseisen kurssin ilmoittautumiseen. Kurssi oli etsittävä itse uudestaan Hellewin Kurssit -sivulta. Tavoite oli saada ilmoittautuminen suoraviivaisemmaksi, ettei käyttäjän tarvitse itse etsiä kurssia kahdesta eri järjestelmästä. Hellewi-järjestelmästä sinänsä ei kuitenkaan ollut tarkoitus luopua. Lisäksi piti selvittää, voisiko esimerkiksi etusivulla listata lähestyvät kurssit, joilla on vielä tilaa.

Sivuille haluttiin myös selkeästi esille sosiaalisen median syötteet, kuten Facebook, Twitter, Instagram ja YouTube. Muita tavoitteita olivat Jyvälän eri toimintojen selkeä

ja yhdenvertainen esille tuominen ja Oiva ja Ilona -toiminnan verkkosivuston yhdistäminen samaan graafiseen ilmeeseen. Lisäksi haluttiin selvittää tarvetta muihin toimintoihin, kuten tapahtuma- tai tilanvarauskalenteriin, uutiskirjeentilaukseen ja palautelomakkeisiin.

Työn päätteeksi oli syytä pitää myös henkilökunnalle perehdytys uuden järjestelmän käyttöönottoon.

4. Työn toteutus

4.1 Vaihtoehtojen karsinta

4.1.1 Yleistä

Sivuston modernisointia aloittaessa piti miettiä vaihtoehtoja, mihin suuntaan sen tekniikkaa ja rakennetta on syytä kehittää. Ensisijaisen tärkeää oli uudistaa sivun ulkoasu ja rakenne niin, että se palvelee käyttäjiä mahdollisimman hyvin, ja samalla tehdä sivuston päivittäminen mahdollisimman selkeäksi. Tärkeää oli myös se, että oppimiskynnys sisällönhallintaan olisi matala, koska sivustoa päivittävien henkilöiden vaihtuvuus on suhteellisen suuri, eikä kokemusta verkkopalvelujen ylläpidosta tai kehittämisestä välttämättä ole. Sivuston päivitystarve keskittyy lähinnä sisältöön, eikä suurempiin sivuston rakenteen muutoksiin ole oletettavasti usein tarvetta. Olisi kuitenkin hyvä, jos pieniä muutoksia olisi tarvittaessa mahdollista tehdä ilman ulkopuolista apua.

Näillä kriteereillä täysin HTML-pohjainen ratkaisu jäi pois vaihtoehdoista, itse koodatun sivun hallinnointi ei onnistu ilman HTML-tuntemusta. Sovelluskehyksen käyttö olisi voinut tulla kysymykseen, jos siihen olisi liitetty CMS-tyylinen sisällönhallinnan käyttöliittymä. Sovelluskehyksen etuna olisi saatu rajoittamattomat jatkokehitysmahdollisuudet (Kallio 2013). Jyvälän Setlementin sivuston kokoisessa sivustossa ei kuitenkaan ole odotettavissa niin suuria rakennemuutoksia, että siitä saataisiin vastaavaa hyötyä. Todennäköisesti sovelluskehystyyppisen sivuston rakenteiden

päivittäminen olisi hieman vaativampaa asiaan perehtymättömälle kuin sisällönhallintajärjestelmän kautta.

Loppupäätelmänä oli, että sivustolle asetetut kriteerit pystyttäisiin varmasti saavuttamaan jonkin sisällönhallintajärjestelmän kautta, ja sen asettamat rajat tuskin tulisivat jatkossakaan vastaan tämän tyyppisessä ja kokoisessa sivustossa. Ja kun haluttiin optimoida myös asiaan perehtymättömän oppimiskynnys, oli sisällönhallintajärjestelmän käyttäminen varmin vaihtoehto.

4.1.2 Käytetyimmät sisällönhallintajärjestelmät

Julkaisujärjestelmien kirjo on nykypäivänä suuri, ja näistä sopivien karsiminen on haastavaa. Kaikkien vaihtoehtojen testaaminen on mahdotonta, joten vaihtoehtoja on karsittava joillakin kriteereillä. Helppokäyttöisyys voi olla yksi valintakriteeri. Erityisesti jos aiempaa ohjelmointi- tai ulkoasusuunnittelukokemusta ei ole, on hyvä, jos järjestelmä tarjoaa mahdollisuuden sivuston yksinkertaiseen mukauttamiseen. Tässä on avuksi, jos järjestelmä tarjoaa valmiit mallipohjat, joita käyttäen voi helposti luoda sisältöä, eikä layoutin suunnitteluun tarvitse käyttää aikaa. Nykypäivän sivuston on mukauduttava myös mobiililaitteeseen. Skaalautuvuus, sivuston latausnopeus ja muutenkin korkea suorituskyky on tärkeää. Järjestelmän hakukoneoptimoitavuuksissakin on eroja. Tämä voi olla tärkeä kriteeri, jos halutaan optimoida hakukonenäkyvyys. (Ardourel 2016).

Valintaa tehdessä on perusteltua harkita laajalti käytössä olevista järjestelmistä, koska niistä löytyy hyvin dokumentaatiota ja käyttäjäkokemuksia. Tämän hetken ylivoimaisesti käytetyin sisällönhallintajärjestelmä on wappalyzer.com-sivuston mittausten mukaan Wordpress, jota käyttää 77 % kaikista julkaisujärjestelmää käyttävistä sivustoista. Seuraavat ovat Joomla 7,5 %:illa ja kolmantena Drupal (Wappalyzer n.d). W3techs.com-sivuston mukaan Wordpressin kannatus on 59,9 % ja seuraavana tulevat Joomla! 6,1 % ja Drupal 4,1 % (Usage of content management systems for websites n.d). Samat kolme ensimmäistä nimeää myös esimerkiksi websitesetup.org, mutta näitä seuraavat sijat vaihtelevat tutkimuksesta riippuen (Mening 2015).

Tutkimuksen aluksi karsittiin pois muut, vähemmän käytetyt julkaisujärjestelmät, ja jätettiin nämä kolme suosituinta. Tutkimukseen haluttiin kuitenkin myös joitakin vähemmän tunnettuja vaihtoehtoja, joiden käyttöön saattaisi löytyä muita perusteita. Yksi tällainen järjestelmä on Couch CMS. Myös nykyisen CMS Made Simple (CMSMS) -järjestelmän päivittäminen oli vaihtoehto.

4.1.3 Järjestelmävaatimukset

Kaikki tutkimukseen valitut sisällönhallintajärjestelmät vaativat PHP:n sekä jonkin tietokantapalvelun. Kaikissa suositeltavinta on käyttää MySQL:ää. Yleisin käytetty palvelinohjelmisto on Apache. Taulukossa 1 näkyy tarkemmin järjestelmävaatimukset sekä kirjoitushetkellä uusin järjestelmäversio. Kaikki tutkittavat järjestelmät ovat avoimen lähdekoodin ohjelmistoja.

Taulukko 1. Vertailtujen sisällönhallintajärjestelmien vaatimukset ja suositukset (muokattu lähteistä Requirements n.d; Drupal 8 system requirements n.d; Installing Joomla n.d; Couch CMS Requirements n.d; CMS Made Simple Requirements n.d).

CMS	**WordPress**	**Joomla**	**Drupal**	**Couch CMS**	**CMSMS**
Versio	4.9.4	3.8	8	2.0	2.0
PHP	5.2.4+, suositeltu 7.2+	5.3.10+, suositeltu 5.6 tai 7+	5.5.9+, suositeltu 7.2	5.0.0+	5.4.11+
Tietokanta	MySQL 5.0+ (suos. 5.6+)/ MariaDB 10.0+	MySQL 5.1+ (suos. 5.5.3+) SQL Server 10.50.1600.1+ PostgreSQL 8.3.18+ (suos. 9.1+)	MySQL 5.5.3 / MariaDB 5.5.20 / Percona Server 5.5.8 + InnoDB ja PDO -lisäosa (suositeltu) PostgreSQL SQLite MongoDB ja MS SQL Server (lisämoduuleilla)	MySQL 4.1.2+	MySQL 4.1+
Palvelinohjelmisto	Apache / Nginx suositeltu, muukin PHP/MySQL -kykyinen käy	Apache 2.x+, Nginx 1.0+, Microsoft IIS	Mikä vain, joka täyttää PHP-vaatimukset	Apache tai Apache-yhteensopiva	Apache 1.3 tai 2, LightTPD 1.4+

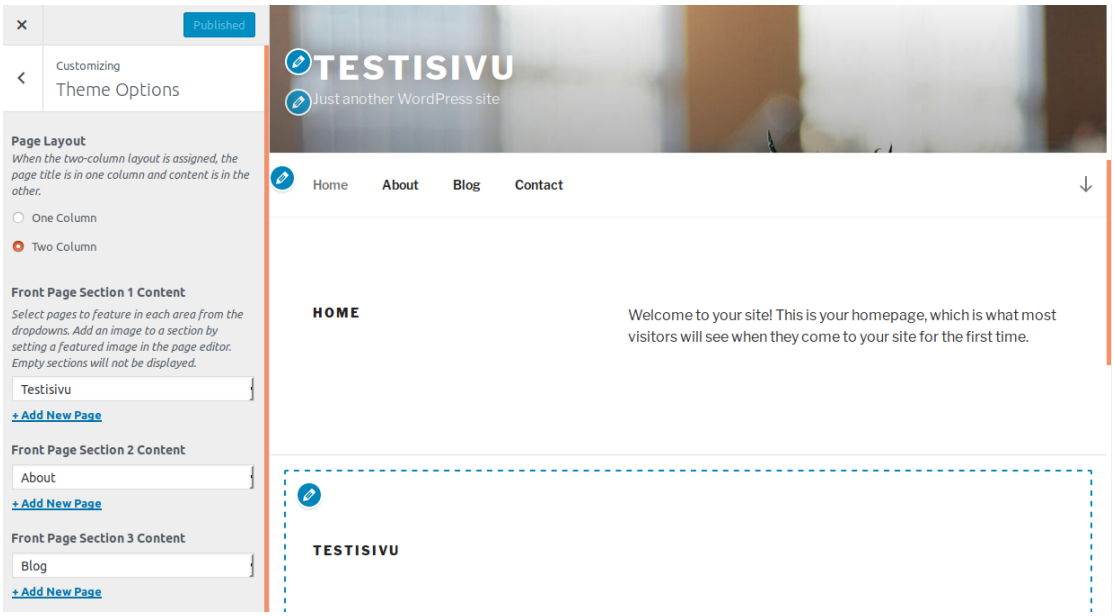
Tutkimuksessa asennettiin WordPress, Drupal, Joomla ja Couch CMS Ubuntu-käyttöjärjestelmään ja tutustuttiin niiden perustoimintoihin. Käytössä oli palvelinohjelmana Apache 2.4.27, PHP versio 7.1.11 ja tietokantaohjelmana MySQL 5.7.21. CMS Made Simple -järjestelmään oli mahdollista tutustua Jyvälän palvelimella.

4.1.4 WordPress

WordPress on saanut alkunsa blogiohjelmistona, mutta laajentunut kokonaiseksi sisällönhallintajärjestelmäksi. WordPressiä, Joomlaa ja Drupalia verrataan usein toisiinsa näiden ollessa kolme suosituinta vaihtoehtoa. WordPressiä pidetään lähes poikkeuksetta helpoimpana käyttää. Muun muassa WebsiteSetup.org pitää WordPressiä parhaana pienille ja keskisuurille verkkosivustoille ja blogeille, Joomlaa suositellaan verkkokauppa-palveluiden pohjaksi ja Drupalia pidetään vaikeimpana, mutta tehokkaimpana. WordPressiin löytyy myös eniten teemoja ja lisäosia. (Mening 2015). WordPress pohjautuu PHP-kieleen ja JavaScriptiin, ja jos sivuston rakentamisessa on päästävä valmiita sivustopohjia ja lisäosia eli plugineita, syvemmälle, voi omaa koodia sisällyttää kyseisillä kielillä (Drupal vs WordPress CMS Comparison Which To Choose? 2017). Tähän julkaisujärjestelmään luottavat muun muassa Sylvester Stallone, Mercedes-Benz, Samsung ja Sony Music (Schäferhoff 2015). Myös Suomessa sitä käytetään laajasti. Jo pelkästään yhden WordPress-pohjaisia sivuja valmistavan yrityksen, Aucorin sivuilta, osoitteesta aucor.fi, löytyy pitkä lista sen pohjalle rakennettuja palveluita. Näitä ovat muun muassa Nokian jalkineet, HYKS ja Joutsenmerkki.

WordPressin asennus aloitetaan luomalla sitä varten tietokanta, purkamalla zip-paketti haluttuun kansioon ja avaamalla sen sijainti selaimella. Asennusohjelma kysyy tietokannan nimen, käyttäjätunnuksen ja salasanan, ja sen jälkeen on asetettava sivuston otsikko, käyttäjänimi ja salasana. Tämän jälkeen eteen avautuu WordPressin ohjauspaneeli, ja sivusto on valmis muokattavaksi. Jo tässä vaiheessa sivu on näyttävän näköinen. Helpoiten ulkoasua pääsee muokkaamaan klikkaamalla Customize this site, tai suomenkielisessä versiossa Tee sivustostasi yksilöllinen, -nappia. Kuviossa 2 on ohjauspaneelin ulkoasu muokkausvaiheessa. Muokkaustapa on hyvin visuaalinen,

sivun osioita pääsee muokkaamaan klikkaamalla kynäikoneita tai klikkaamalla sivun osion päällä shift-näppäin pohjassa. Muokkausvalikko nousee tällöin sivun vasempaan reunaan. Yläreunan Publish-nappia klikkaamalla muutokset tulevat käyttöön. Ohjauspaneelin sivuvalikosta Pages-kohdasta voi luoda uusia sivuja, ja Appearance-painikkeesta mm. sijoittaa sivut valikoiden alle, asentaa uusia teemoja, lisätä sivulle vimpaimia ja vaihtaa tausta- tai otsakekuvaa ynnä muuta. WordPressin asennus ja käyttöönotto on varsin vaivatonta, näyttäviä teemoja on helppo asentaa ja niitä on runsaasti tarjolla. Näin varsinaista sisältöä pääsee tuottamaan sivuille hyvinkin nopeasti.



Kuvio 2. WordPress-sivun muokkaus

4.1.5 Drupal

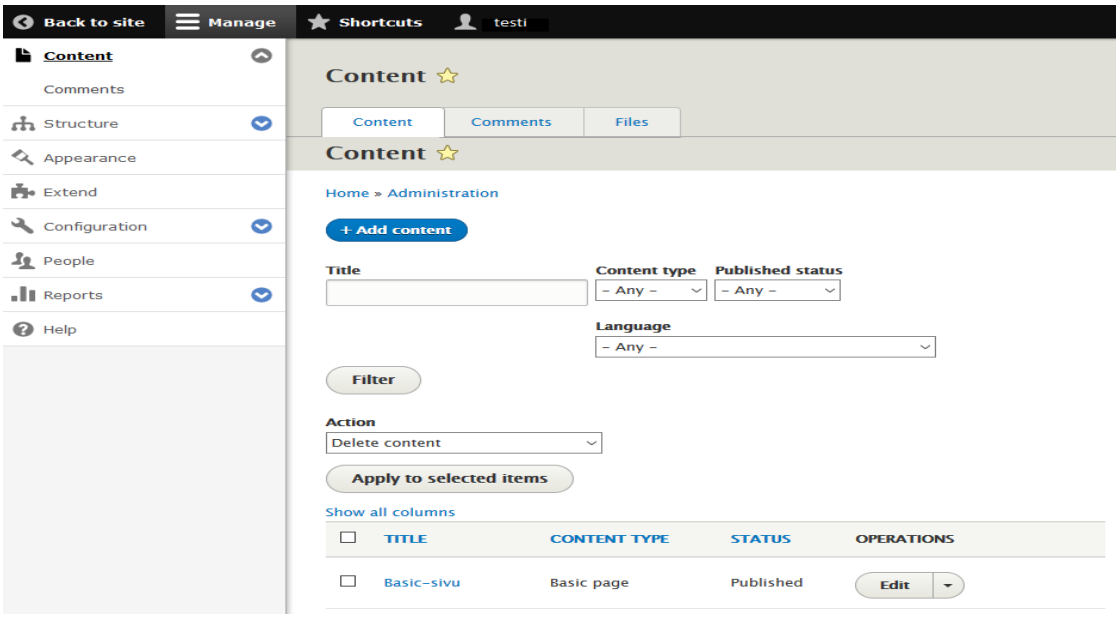
Myös Drupalilla on internetissä vankka kannattajajoukkonsa, mutta WordPressiin verrattuna sen suosiota on haitannut korkeampi oppimiskynnys. Esimerkiksi suomalainen Kukumo-mainostoimisto kertoo käyttävänsä Drupalia lähes kaikissa projekteissaan, mutta kuvailee sen olevan lähempänä sovelluskehystä kuin julkaisujärjestelmää. Toisaalta siis Drupalin etu WordPressiin verrattuna on sen suurempi

muokattavuus, mutta toisaalta käyttöönotto ei ole niin yksinkertaista. Drupal koostuu moduuleista, joita käyttäen sivustoista voidaan rakentaa hyvin erityyppisiä kokonaisuuksia. (Miksi me käytämme Drupalia? n.d.)

Tolppi (2013, 14) kertoo opinnäytetyössään, että Drupal on käytössä muun muassa YLE:llä, Suomen palloliitolla ja Valkoisella talolla. Netistä löytyy paljon vertailuja Drupalin ja WordPressin välillä, joissa vahvistetaan, että Drupalin oppimiskynnys on jyrkempi. Myös käyttöönotto- ja ylläpitokustannuksia kuvaillaan selvästi suuremmiksi Drupalin osalta mm. kasareviews.com -sivustolla, jossa kerrotaan myös, että Drupalin käyttöönottoa harkitsevan on syytä olla perillä myös HTML- ja PHP-kielten perusteista (Drupal vs WordPress CMS Comparison Which To Choose? 2017).

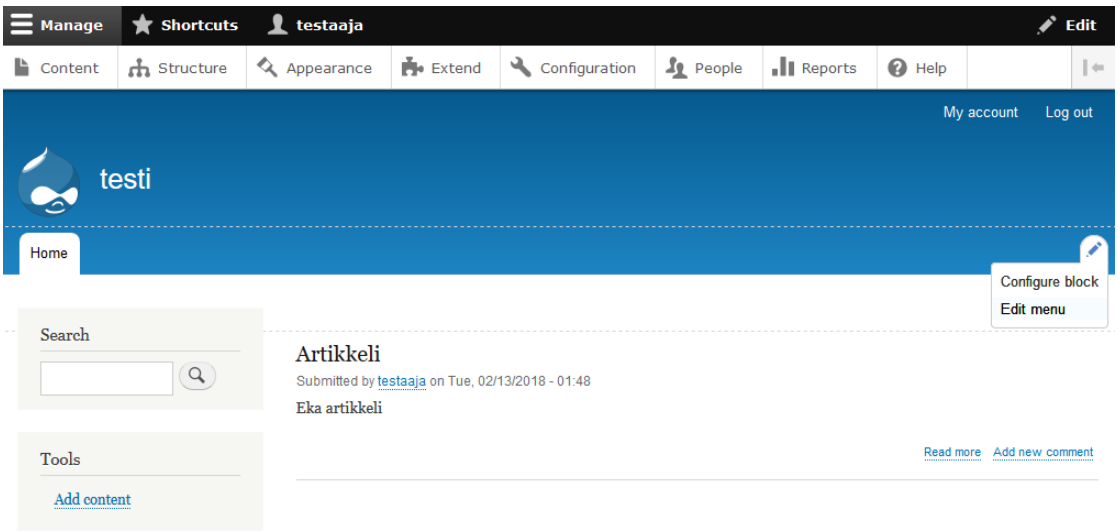
Drupalin asennus alkaa purkamalla paketti haluttuun kansioon, jonka jälkeen se jatkuu selaimen kautta palvelimen osoitteesta. Jos käyttöön halutaan suomen kieli, on Drupalin sites/default/-kansioon luotava hakemistorakenne files/translations/ ja haettava käännöspaketti osoitteesta https://localize.drupal.org/. Asennusvaihtoehtoina on Standard, joka asentaa yleisimmät toiminnot, sekä Minimal, joka antaa käyttäjän määritellä tarkemmin asennettavat osat. Sites -kansioon on tehtävä kopio default.settings.php-tiedostosta nimellä settings.php. Default.settings.php:ta ei saa uudelleennimetä settings.php:ksi, vaan Drupal vaatii myös sen olemassaolon. Sites-kansion alahakemistoineen on asennuksen aikana oltava kirjoitussuojaamaton. Tässä vaiheessa asennus antoi virheen “Drupal requires you to enable the PHP extensions”, vaikka php7.1-gd ja php7.1-mysql oli asennettu. Virhe korjaantui asentamalla myös phpmyadmin. Netin keskustelupalstojen mukaan vastaavaa ongelmaa on ollut paljonkin, ja se on mahdollista korjata eri tavoin. Seuraavassa vaiheessa annetaan käytettävän tietokannan nimi, käyttäjätunnus ja salasana. Asennusskripti asentaa tämän jälkeen tarvittavat osat. Lopuksi ilmoitetaan, että Drupal on asennettu, ja kirjoitussuojaus on syytä palauttaa sites-kansioon. Tästä eteenpäin mikään toiminto ei kuitenkaan toiminut, vaan selain antoi virheilmoitusta “the requested url node/add was not found on this server”. Tämä ongelma ratkesi lisäämällä /etc/apache2/sites-available/000-default.conf -tiedostoon rivit:
<Directory /var/www> AllowOverride all </Directory>

Ensinäkemältä Drupalin käyttöliittymä (kuvio 3) vaikuttaa yksinkertaiselta, mutta silti on hieman vaikea hahmottaa, mistä sivuston rakentaminen kannattaa aloittaa.



Kuvio 3. Drupal-käyttöliittymä.

Kuviossa 4 näkyy sivusto ensimmäisen artikkelin lisäämisen jälkeen. Teemana on Bartik, joka on visuaalisin mukana tulevista teemoista. Teemaa voidaan vaihtaa ylävalikon Appearance-kohdasta. Sivu näyttää melko karulta ja tässä vaiheessa on helppo uskoa, että Drupal-sivuston käyttövalmiiksi saattaminen ei suju yhtä nopeasti kuin esimerkiksi WordPressillä.



Kuvio 4. Drupal-testisivu.

Sivustolle saadaan sisältöä lisäämällä Add content -napin (tai ylävalikon content) kautta uusia sivuja ja artikkeleita. Uusille sivuille saadaan valikkoon omat välilehdet klikkaamalla valikko-osion kynäkuvaketta, ja sen Edit menu -vaihtoehdon alta löytyvää Add link -nappia. Ylävalikon Structure-kohdasta voidaan määritellä sivun eri osiin erilaisia blokkeja. Esimerkiksi valikon paikan voi siirtää vaikkapa sivupalkkiin header-osion sijaan, tai vaihtaa hakukentän sijaintia ja lisätä yhteydenottolomakkeen, muutamia mainitakseni. Extendistä voidaan lisätä moduuleita. Moduuleina löytyy muun muassa bannerien hallinta, chat-palvelu, gallupit, keskustelufoorumit, Google Analytics, verkkokauppa ja tapahtumakalenteri (Drupal-julkaisujärjestelmä n.d.). Ulkoasua voi muokata eri tavoin teemasta riippuen Appearance-kohdasta. Bartik-teemassa ei voida muuttaa juuri muuta kuin värejä ja logoa. Uusia teemoja on toki saatavilla. Niitä varten on ensin asennettava uusi moduuli, Update Manager, jonka jälkeen Appearance-kohtaan tulee Install new theme -painike. Kun Drupalin hallinnasta pääsee jyvälle, ei perussivuston rakentaminen ole kovin hankalaa, mutta varsinkin visuaalisuuden osalta muokkaaminen ei ole lähellekään niin kätevää kuin WordPressillä.

4.1.6 Joomla

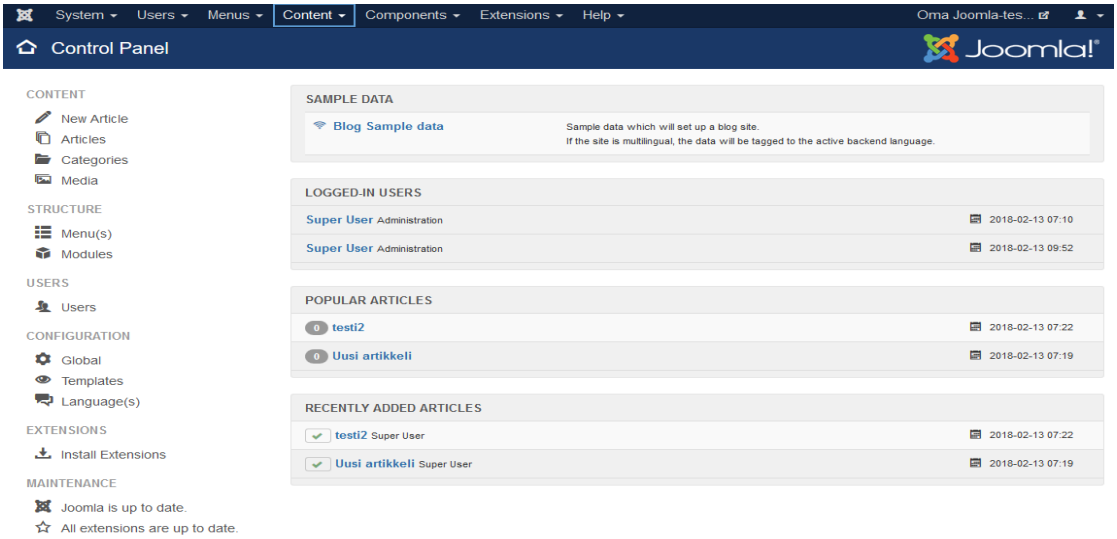
Joomla kilpailee Drupalin kanssa suosiosta maailmalla, ja myös Suomessa sillä on aktiivinen yhteisö takanaan, jonka pääasiallinen sivusto on osoitteessa joomla.fi. Silti on vaikea löytää tietoa Joomlalla toteutetuista tunnetuista suomalaisista verkkosivuista, päinvastoin kuin WordPressin ja Drupalin tapauksessa. Joomlan eduiksi voidaan lukea muun muassa versionhallinta, hakukoneoptimointi ja tuhannet lisäosat (Joomla n.d.). Kuten WordPress ja Drupal, Joomlakin on kirjoitettu PHP-kielellä (Joomla! Documentation n.d).

Joomlan asennus tapahtuu samaan tapaan kuin muillakin testatuilla järjestelmillä. Asennuspaketti puretaan haluttuun kansioon, joka sijaitsee palvelinohjelmistoon asetetun juuren alla. Sen jälkeen Joomla konfiguroidaan selaimen kautta kyseisestä kansiosta. Asetuksissa asetetaan mm. pääkäyttäjän tunnus ja salasana, sekä tietokannan tyyppi. PHP:n tietokantaluokaksi Joomla ehdottaa MySQLi tai MySQL PDO, ja kertoo että käytössä oleva luokka on luultavasti MySQLi. MySQLi toimii nimensä mukaisesti vain MySQL:n kanssa sekä funktio- että olio -pohjaisesti. PDO puolestaan on

oliopohjainen ja toimii MySQL:n lisäksi useiden eri tietokantajärjestelmien kanssa (Doyle 2010, 359). Seuraavaksi asetuksiin on laitettava tietokannan nimi, joka on siis täytynyt luoda esim. MySQL-ohjelman kautta, sekä käyttäjätunnus ja salasana. Tämän jälkeen annetaan mahdollisuus ottaa käyttöön FTP-pohjainen sivuston päivitys, jolle on myös luotava käyttäjätunnus ja salasana. Lopuksi annetaan vielä mahdollisuus asentaa englanninkieliset esimerkkiartikkelit, näytetään yhteenveto asennuksesta sekä listataan Joomlan tarvitsemat järjestelmäasetukset ja kerrotaan, vastaavatko nykyiset asetukset vaatimuksia.

Jos palvelimen konfiguraatio vastaa vaatimuksia, Joomla asentuu Asenna-painikkeesta. Tässä vaiheessa helpommalla pääsee, jos antaa www-data-käyttäjälle oikeudet html-kansioihin ennen Asenna-painikkeen painamista. Tällöin Joomla hoitaa configuration.php:n asetukset kuntoon ja poistaa asennuskansion. Muussa tapauksessa nämä on tehtävä manuaalisesti itse.

Joomlan ohjauspaneelissa (ks. kuvio 5) vasemmassa reunassa on sivupalkki, josta löytyy perustoiminnot. Alkuun pääseminen sivuston muokkaamisessa on kuitenkin selvästi mutkikkaampaa kuin WordPressissä. Äkkiseltään tuntuu oudolta, että vaikka lisää uuden artikkelin, se ei kuitenkaan tule sivulle näkyviin. Joomlaa käyttäessä täytyykin sisäistää oikea järjestys. Ensin on luotava kategoria Categories-painikkeesta, sitten artikkeli, ja artikkelille valittava aikaisemmin luotu kategoria. Menus-painikkeesta päästään lisäämään sivulle valikko, josta aiemmin luotu artikkeli saadaan esille.



Kuvio 5. Joomlan ohjauspaneeli

4.1.7 Couch CMS

Couch CMS on mielenkiintoinen avoimen lähdekoodin sisällönhallintajärjestelmä. Se on kätevä erityisesti siinä tapauksessa, että vanha, staattinen, html-sivu halutaan siirtää sisällönhallinnan alle. Olemassaolevaan html-koodiin lisätään Couch CMS:n omat tagit, joilla määritellään mitä kohtia sivulla päästään muokkaamaan ja millä tavalla. Hallintapaneelin kautta päivityksiä tehdessä ei voi vahingossakaan vahingoittaa sivun rakennetta. Pelkästään sen sisältöä, kuten tekstiä ja kuvia, voi muuttaa, mikäli järjestelmä on rakennettu oikein. (Couch CMS).

Couch CMS vaatii PHP 5.0.0:n MySQL 4.1.2:n Apachen. Käyttöönotto tapahtuu purkamalla zip-paketti haluttuun kansioon, luomalla tietokanta, ja asettamalla config.php -tiedostoon tietokannan käyttäjätunnus ja salasana. Järjestelmää ei ole suomenkielisenä, mutta oleellisimmat kohdat on melko helppo itsekin suomentaa tekemällä Lang-kansion EN.php -tiedostosta kopioi nimellä FI.php, suomentamalla sitä, ja vaihtamalla config.php:hen kieleksi FI.

Couch CMS:n asentamisen jälkeen muokattavien HTML-sivujen alkuun on lisättävä koodi <?php require_once('couch/cms.php'); ?> ja loppuun <?php COUCH::invoke(); ?> Tämän jälkeen määritellään hallintapaneelissa muokattavat koodin osat.

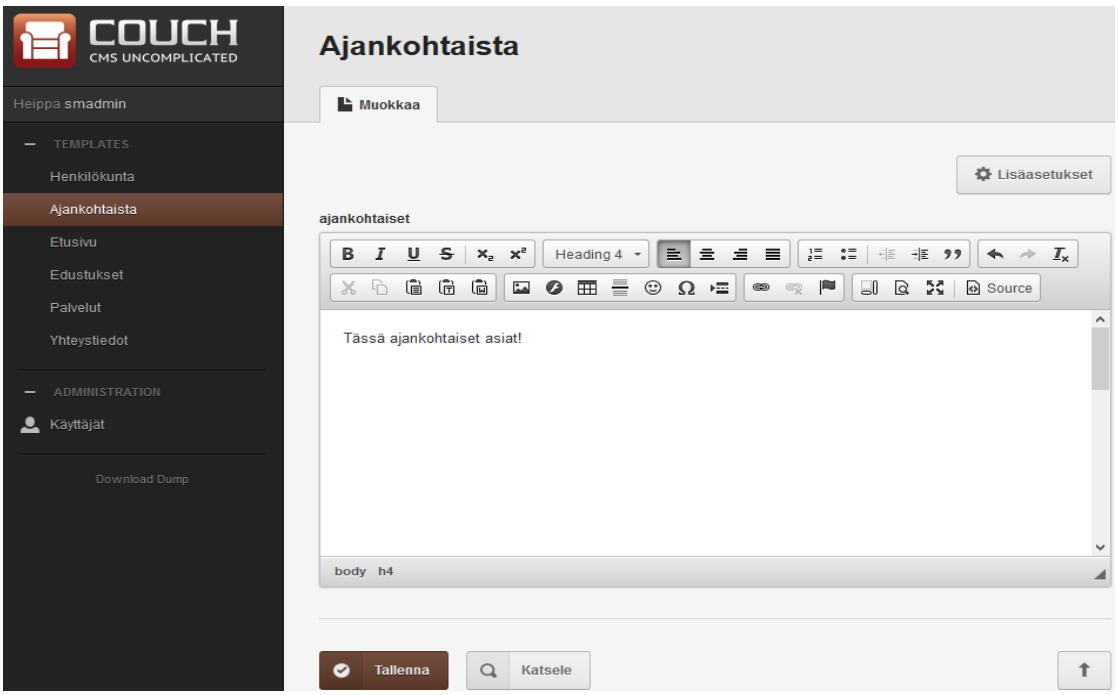
Esimerkki, jossa HTML-sivun alussa luodaan Couch CMS:n "textbox"-tyyppinen kenttä:

<cms:template title='Ajankohtaista'>

<cms:editable name='ajankohtaiset' type='richtext' />

</cms:template>

Vielä on lisättävä rivi *<cms:show ajankohtaiset />* , HTML-koodin kohtaan, jossa kentän sisältö halutaan näkyviin. Näiden muutaman rivin lisäämisen jälkeen sivun ”ajankohtaista” -osio on muokattavissa Couch CMS:n käyttöliittymän kautta, kuten kuvassa 5 näkyy. Käyttöliittymä avautuu vakioasetuksilla selaimeen osoitteella *sivunosoite.verkkotunnus/couch/login.php*.



Kuvio 6. CouchCMS -käyttöliittymä

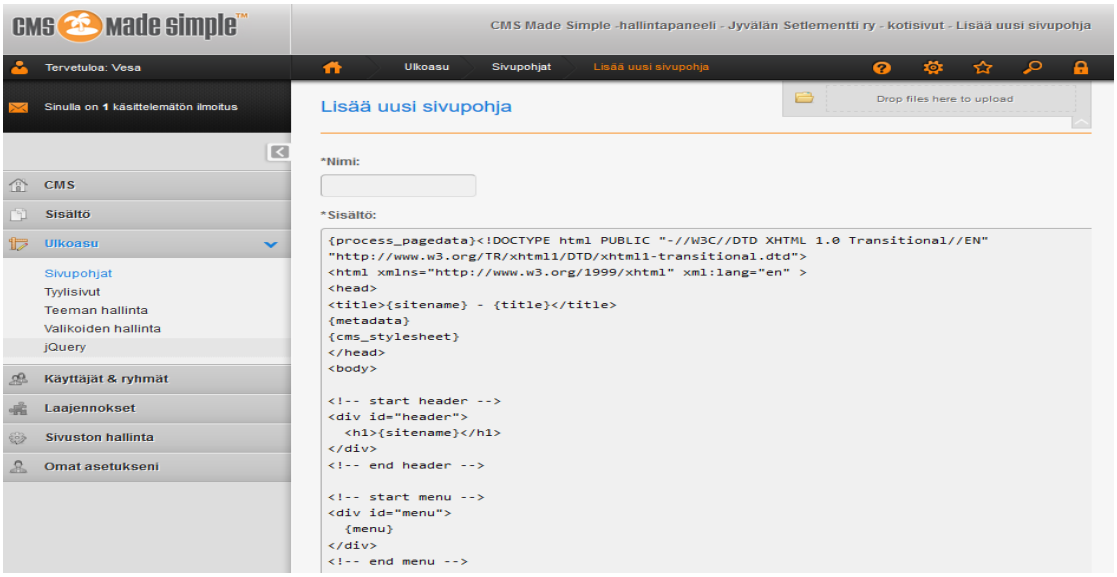
Couch CMS on miellyttävä vaihtoehto, jos nykyinen sivusto on ulkoasultaan ja rakenteeltaan valmiiksi hyvä, mutta sisällönhallintaa halutaan helpottaa. Jyvälän tapauksessa Couch CMS ei kuitenkaan ollut järkevin vaihtoehto, koska sivusto oli syytä uudistaa kokonaan sekä rakenteeltaan että ulkoasultaan.

4.1.8 CMS Made Simple 2.x

Yhtenä vaihtoehtona julkaisujärjestelmän uudistamisessa oli vaihtaa vanha CMS Made Simple uusimpaan 2.1 -versioon. CMS Made Simple -sivuston ohjeiden mukaan suoraa päivittämistä vanhasta uusimpaan versioon ei suositella, vaikka se onkin periaatteessa mahdollista. (Upgrading old CMS Made Simple versions n.d.) CMSMS –sivustolta löytyy ohjeet eri versioiden päivittämiseen. Nykyisestä versiosta riippuen päivitys tapahtuu erilaisten vaiheiden kautta. 1.11.4 -versio on päivitettävä ensin 1.11.3 -versioon, ja siitä 1.12.1 -versioon, joka puolestaan on mahdollista päivittää 2.x -versioksi.

2.x -versiossa on useita parannuksia vanhaan verrattuna. Muun muassa sivupohjien ja tyylisivujen toiminnallisuus on poistettu yksittäisiltä PHP-tiedostoilta ja rakennettu moduuliksi. Modulaarisuus helpottaa muun muassa tietokantojen ja ajax-tekniikoiden käyttöä, ja vuorovaikutusta käyttäjien kanssa. 2.0 -versiossa on myös entistä vähemmän rajoituksia. Tämä sallii suurempien verkkosivustojen luomisen. Myös useampi sivupohja-API on kirjoitettu uudestaan käytännöllisempään muotoon. (Introducing CMSMS 2.0 n.d.).

CMS Made Simple:n ohjauspaneelin kautta voidaan Ulkoasu-kohdasta lisätä uusia sivupohjia ja muokata vanhoja. Sivupohjaa luodessa avautuu tekstilaatikko, johon CMSMS luo automaattisesti html-koodia, jonka sisään sisältöä laitetaan. Pohjaan tulee valmiiksi esimerkiksi koodi otsikon luomiseen: <title>{sitename} – {title}</title>, kuten näkyy kuviossa 7. Sivupohjien muokkaamiseen tarvitaan siis HTML-rakenteen tuntemista.



Kuvio 7. CMS Made Simple, sivupohjan lisääminen

Ulkoasun kautta voidaan muokata myös css-tyylisivuja ja muokata valikkoja. Sisältö-painikkeesta luodaan ja muokataan sivujen sisältöä. Painiketta painaessa avautuu valikko olemassa olevista sivuista (ks. kuvio 8). Sivun otsikkoa painamalla avautuu näkymä, jossa sisältöä voidaan muokata tekstikenttien avulla joko WYSIWYG-tyylisesti tai HTML-muodossa. Ulkoasun muokkaaminen ei ole lähellekään niin helppoa kuin WordPressissä, vaan vaatii HTML- ja CSS -tuntemusta. Helpointa on etsiä mahdollisimman sopiva teema ja asentaa se erillisen ThemeManager-moduulin kautta.

Kuvio 8. CMS Made Simple -ohjauspaneelin näkymä Sisältö-välilehdellä

Parannuksista huolimatta, voidaan sanoa, että toimeksiantaja oli kyllästynyt CMS Made Simple järjestelmään, eikä uudemman version uskottu tuovan tähän muutosta. Suurimmat muutokset järjestelmässä ovat “pinnan alla”, eivätkä niinkään käyttäjän havaitsemina parannuksina. Toisaalta sivut haluttiin ulkoasun puolesta rakentaa uudestaan ja siksi vanhan sivuston muuntaminen uudemmalle alustalle ei olisi ollut käytännöllistä.

4.2 Julkaisujärjestelmän valinta

Järjestelmiä vertailtaessa Couch CMS ja CMS Made Simple tippuivat heti pois vaihtoehdoista. Couch CMS olisi sallinut vain rajatun sisällön muokkaamisen sisällönhallinnan kautta. Ulkoasuun liittyvät muutokset olisivat vaatineet HTML-koodauksen käyttöä. Myöskään nykyisen CMSMS-järjestelmän päivittämisessä uudempaan versioon ei nähty varsinaista hyötyä. Sivut oli syytä joka tapauksessa rakentaa uudestaan, eikä uudempi versio näyttänyt tarjoavan merkittävää helpotusta sivujen ylläpitoon vanhempaan verrattuna. Toimeksiantajallakin oli kyseisestä järjestelmästä syntynyt melko negatiivinen kuva.

Sisällönhallintajärjestelmän valinta tehtiin Drupalin, Joomlan ja Wordpressin kesken. Vaikka asennuksen helppoudella ei ollut varsinaisen työn kannalta merkitystä, antoi jo asennusvaihe viitteen siitä, että Wordpress olisi näistä helpoin vaihtoehto. Ja järjestelmien toimintoja vertaillessa tämä oletus vahvistui. Esimerkiksi artikkelien lisääminen ja visuaalisen näkymän muokkaus oli WordPressissä selvästi helpointa. Tehdyt muutokset olivat myös helpoiten heti todettavissa. Sisällönhallinnan kannalta WordPress on vertailun perusteella helppokäyttöisin vaihtoehto asiaan vihkiytymättömälle, mikä oli tässä työssä tärkeä kriteeri. Myös Meningin (2015) mukaan se on paras vaihtoehto aloittelijalle ja sopii erityisen hyvin pienille ja keskisuurille sivustoille. WordPressiin löytyy myös eniten helposti käyttöönotettavia lisäosia. Toisaalta maailman suosituimpana julkaisujärjestelmänä siihen löytyy myös valtavat määrät ohjeita, joista sivuja päivittävä voi saada apua. Näillä perusteilla päädyttiin toteuttamaan sivusto Wordpressin pohjalle.

5. Sivuston toteutus WordPressillä

5.1 Yleistä

Kun oli selvää, että työ toteutetaan WordPress-alustalle, tultiin Jyvälän viestintätiimin kanssa siihen tulokseen, että WordPressin Karuna-teema on ulkoasultaan ja rakenteeltaan melko lähellä sitä, jollaiseksi sivut haluttiin. Sivuston ja valikkojen rakennetta alettiin näin suunnittelemaan sen mukaan.

5.2 Lapsiteema

Vaihtoehtona oli Karuna-tyylisen oman teeman toteutus, mutta teema päädyttiin tekemään niin sanottuna lapsiteemana. Yleisesti ottaen oma teema kannattanee tehdä vain, jos ei ole olemassa valmista teemaa, joka kävisi ilman muutoksia tai lapsiteemaksi muokattuna kyseiseen käyttötarkoitukseen. Omaa teemaa voisi käyttää esimerkiksi pienimuotoisessa blogisivussa, josta haluaisi täysin persoonallisen tai toisaalta oma teema voisi olla tarpeen, jos sivusto on niin suuri ja monimutkainen, että valmiin teeman käyttöönotto vaatisi liikaa muutostöitä.

Valmiin teeman asentaminen ja sen koodin muokkaaminen omaan käyttöön saattaa myös vaikuttaa hyvältä vaihtoehdolta, mitä se ei kuitenkaan ole. Jos asennettuun teemaan tulee esimerkiksi tietoturvaan liittyvä päivitys, omat muutokset koodista voivat hävitä. Lapsiteeman käytössä sen sijaan on useita etuja. Teeman päivitys ei myöskään aiheuta ongelmia, ja kehitystyö on huomattavasti nopeampaa, kun saadaan heti käyttöön valmiin teeman testatut ominaisuudet, tyylit ja sivupohjat. Oman teeman koodaamisessa on otettava huomioon kaikki mahdolliset skenaariot ja sovitettava koodi niihin, mutta lapsiteemaa käyttäessä ominaisuudet ovat olemassa vaikkei niitä erikseen tekisi. Esimerkkinä mobiilikäyttö; lapsiteema sisältää mobiiliominaisuudet ja responsiivisuuden vaikkei niitä erikseen koodata, koska ne periytyvät isäntäteemalta. (What is a WordPress Child Theme? Pros, Cons and More 2013).

Koska lapsiteema perii isäntäteemalta ominaisuutensa, voidaan lapsiteemaan kätevästi tehdä muutoksia vain haluttuihin kohtiin, ja sivusto toimii hyvin vaikkei kaikkea

erikseen koodata. Käytännössä teemalle luodaan oma kansio ja jos kansio sisältää template-tiedostoja, kuten esimerkiksi header.php, niitä käytetään isäntäteeman vastaavan tiedoston sijaan. Muussa tapauksessa käytetään isäntäteeman tiedostoja. Teemalle voidaan luoda myös oma CSS-tyylitiedosto, jossa olevat määritykset ohittavat isäntäteeman vastaavat, sekä functions.php, johon luodaan lapsiteeman luokat ja funktiot. Lapsiteeman functions.php latautuu WordPressissä ensin ja heti sen perään isäntäteeman functions.php. (Child Themes n.d.).

5.3 Koukut

WordPressissä on kahden tyyppisiä koukkuja, englanniksi hooks, jotka tarkoittavat WordPressin suorituksen aikana tapahtuvia tilanteita, joihin voidaan tarttua ja suorittaa kyseisessä tilanteessa omaa koodia. Näin olemassa olevaan koodiin voidaan tehdä lisäyksiä ja muutoksia, ilman että tosiasiassa muutetaan alkuperäistä koodia. Action hooks -koukuilla, joiden funktio on add_action(), voidaan omaa koodia lisätä. Filter hooks-koukuilla, funktiolla add_filter() ,voidaan manipuloida sisältöä. Teemassa näitä koukkuja on valmiina tietyissä kohdissa, esimerkiksi widgets_init -koukku ajetaan aina vimpaimia alustettaessa, mutta niitä voi luoda myös itse. (Poranen 2015, 4-6).

5.4 JQuery WordPressissä

WordPress sisältää jQuery-paketin, joten sitä ei tarvitse erikseen asentaa. Omat jQuery-skriptit kannattaa lisätä sivuille add_action('wp_enqueue_scripts') -koukkuun. Add_actionin toisena parametrina annetaan skriptin rekisteröintifunktio, jossa skripti puolestaan rekisteröidään funktiolla wp_register_script (), ja asetetaan tarvittaessa ladattavaksi wp_enqueue_script () -funktiolla. WordPressin mukana toimitettavan jQueryn erikoisuus on, että normaalisti käytetyn $-merkin sijaan koodin alussa käytetään sanaa jQuery.

5.5 Tietoturvanäkökohtia

WordPressin asennuksessa pätevät monet samat tietoturvasäännöt kuin muuallakin netissä, kuten helppojen käyttäjänimien ja salasanojen välttäminen. Esimerkiksi

yleistä ’admin’ -käyttäjänimeä kannattaa välttää. Lisäksi WordPressin vakioasetus tietokannan etuliitteeksi on wp_. Tämä antaa hakkereille mahdollisuuden kohdistaa hyökkäyksiä tällaisia tietokanta-etuliitteitä sisältäviin sivuihin, ja siksi sekin kannattaa vaihtaa WordPressiä asentaessa vähemmän tavalliseksi. Kannattaa myös huolehtia WordPressin ja käytössä olevan teeman päivityksien pitämisestä ajan tasalla.

Myös wp-config.php:n kautta tietoturvaa voi parantaa. Tiedoston ’.htaccess’ kautta voidaan wp-config.php piilottaa näkyvistä määrityksellä ’deny from all’. Wp-config.php:hen voidaan myös lisätä rivi define('DISALLOW_FILE_EDIT', true), joka estää WordPressin tiedostojen muokkaamisen hallintapaneelin kautta. Jos pääkäyttäjänä onnistuttaisiinkin murtautumaan sisään, ei PHP-tiedostoja voitaisi sitä kautta muokata. Tietoturvan parantamiseksi on myös plugineita, muun muassa sellaisia, joilla voidaan rajoittaa sisäänkirjautumisyrityksiä. (Heijmans 2018).

Plugineita tehdessä on hyvä estää näiden PHP-tiedostojen suora käyttö. Tähän on useita tapoja, mutta yksi helppo tapa on laittaa tiedoston alkuun rivi defined('ABSPATH') or die (''). Tällä tarkistetaan, että ABSPATH-muuttuja on määritelty ennen PHP-tiedoston ajamista. (Writing a Plugin n.d.).

Plugin-lomakkeiden yhteydessä kannattaa käyttää WordPressin sanitize_text_field () -funktiota, joka tarkistaa syötetyn kentän formaatin, ja mm. poistaa html-tagit ja rivinvaihdot. Myös käyttäjän käyttöoikeuksien tarkistamiseen on useita funktioita, kuten current_user_can() - ja is_admin () -funktiot.

Tietoturvaa voidaan lisätä myös käyttämällä WordPressin erityisiä nonce-funktioita, jotka liittävät eräänlaisen käyttäjäkohtaisen varmenteen haluttuun tapahtumaan. Varmenne on voimassa 24 tuntia. Esimerkiksi lomakkeeseen voidaan liittää wp_nonce_field () -funktio. Tämä liittää lomakkeen lähetystapahtumaan tarkistussumman, joka voidaan varmentaa ennen tietokannan päivittämistä esimerkiksi check_admin_referer () -funktiolla. Noncella autentikoimattomilta käyttäjiltä evätään lomakkeiden ja urlien väärinkäyttö. (Petreski 2015).

5.6 Jyvälän teeman toteutus

5.6.1 Lapsiteeman luominen

Lapsiteeman tekeminen aloitettiin luomalla WordPressin themes-kansioon jyvala-niminen kansio. Lisäksi piti luoda CSS-tiedosto, ks. kuvio 9, jonka alkuun oli kommentteihin laitettava tietoja teemasta. Välttämättömät tiedot ovat lapsiteeman nimi sekä teema, josta se periytyy.

```
/*
Theme Name:   Jyvala Theme
 Theme URI:
 Description:  Jyvalän WP-teema
 Author:       Vesa Vertainen
 Author URI:
 Template:     karuna
 Version:      1.0.0
 License:      GNU General Public License v2 or later
 License URI:  http://www.gnu.org/licenses/gpl-2.0.html
 Tags:         responsive-layout
 Text Domain:  karuna-child
*/
```

Kuvio 9. Lapsiteeman CSS-tiedoston otsikko-osa

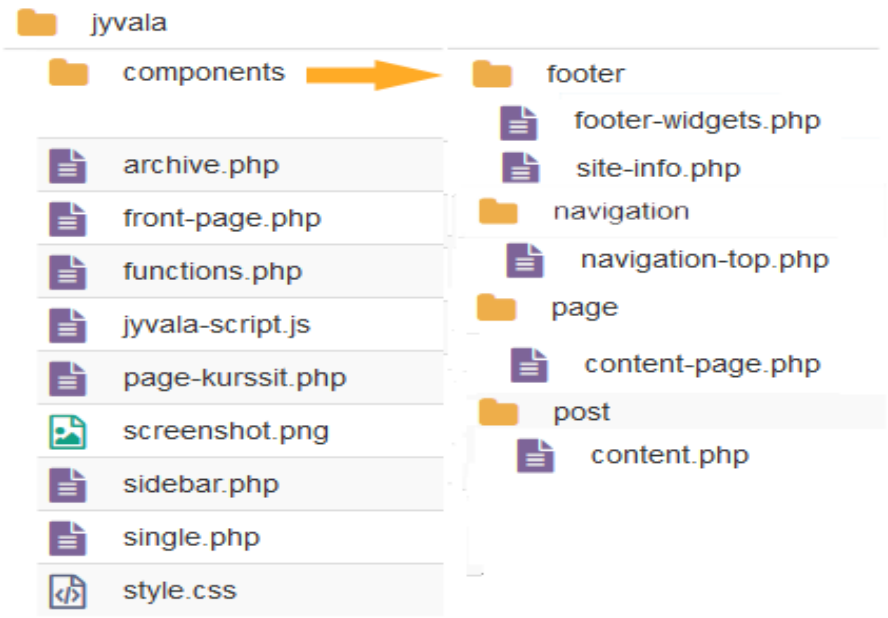
Myös functions.php-tiedosto oli luotava jyvala-kansioon. Tässä vaiheessa lapsiteema oli jo käyttöönotettavissa WordPressin ohjauspaneelista, mutta ilman toiminnallisuuksia. Lapsiteeman CSS-tiedoston lisäksi, pohjalle otettiin käyttöön isäntäteeman CSS. Tällöin kaikki isäntäteeman tyylit olivat käytössä, ja lapsiteeman CSS:n kautta voitiin uudelleen muotoilla tarvittavia kohtia. Isäntäteeman CSS-tyylin voi ottaa käyttöön lapsiteeman CSS-tiedostossa komennolla @import url(../parent-theme/style.css), mutta on suositeltavampaa tehdä se functions.php:n kautta. CSS otettiin käyttöön WordPressin enqueue-funktiolla, kuten näkyy kuviossa 10. Jyvala-styles-funktiossa ladataan ensin isäntäteeman, sitten lapsiteeman CSS. Itse funktio liitetään wp_enqueue_scripts-koukkuun.

```
add_action( 'wp_enqueue_scripts', 'jyvala_styles' );

function jyvala_styles() {
   $parent_style = 'karuna-style';

   wp_enqueue_style( $parent_style, get_template_directory_uri() . '/style.css' );
   wp_enqueue_style('child-style',  get_stylesheet_directory_uri() . '/style.css',
    array( $parent_style ),
    wp_get_theme()->get('Version')
   );
```

Kuvio 10. CSS-tiedostojen liittäminen lapsiteeman functions.php:ssa

Teeman värit, logo ja taustakuva haluttiin Suomen setlementtiliiton ohjeistuksen mukaisiksi. Taustakuva ja logo voitiin muuttaa WordPressin ohjauspaneelin kautta, mutta värimaailma määriteltiin lapsiteeman CSS-tiedostoon. Käytännössä oli käytävä läpi isäntäteeman CSS-tiedostosta läpi kohdat, joissa värejä oli määritelty ja luotava lapsiteemaan sama määrittely, mutta ohjeistuksen mukaisilla väreillä. Esimerkiksi valikoiden taustaväri oli määritelty isäntäteemassa valkoiseksi, lapsiteemassa se ylikirjoitettiin oranssiksi määrittelyllä *.main-navigation ul ul { background-color: #FF6D22; }*.

Seuraavaksi alettiin käymään läpi sivupohjia, joihin haluttiin muutoksia. Muokattava php-tiedosto kopioitiin isäntäteemasta lapsiteeman kansioon ja tehtiin siihen halutut muutokset. Kuviossa 11 näkyy valmiin teeman kansiorakenne ja muokatut sivupohjat. Screenshot.png on kuvakaappaus valmiin teeman etusivusta, joka näkyy teemaa valitessa WordPressin ohjauspaneelissa.



Kuvio 11. Jyvälä-teeman kansiorakenne

5.6.2 Etusivu ja arkistosivu

WordPressin asetuksista etusivuksi voi asettaa viimeisimmät artikkelit, tai staattisen etusivun. Jyvälän aloitussivulle ei haluttu blogia vaan muokattava tekstikenttä, johon voidaan itse manuaalisesti laittaa esimerkiksi uusimpia tapahtumia tai tiedotteita. Sivu asetettiin staattiseksi ja tällöin siihen kohdistuvat muutokset tehtiin lapsiteeman juureen luotuun front-page.php:hen. Sivupohjaan lisättiin ainoastaan suurehko otsikkoteksti ja muokattavan tekstikentän alle linkki artikkelien listaukseen eli arkistoon. Linkin koodaus näkyy kuviossa 12. Arkiston sivupohja puolestaan on archive.php-tiedostossa, joka kopioitiin sellaisenaan ainoastaan Category-otsikko käytöstä poistaen.

```
<?php
// haetaan Arkisto-sivun id-numero nimen perusteella
$page = get_page_by_title( 'Arkisto');

// asetetaan id-numero linkiksi  ?>
<a href="<?php echo '?page_id='.$page->ID; ?>"><p>Lue lisää artikkeleita...</p></a>
```

Kuvio 12. Arkistolinkin lisäys etusivulle

5.6.3 Perussivut

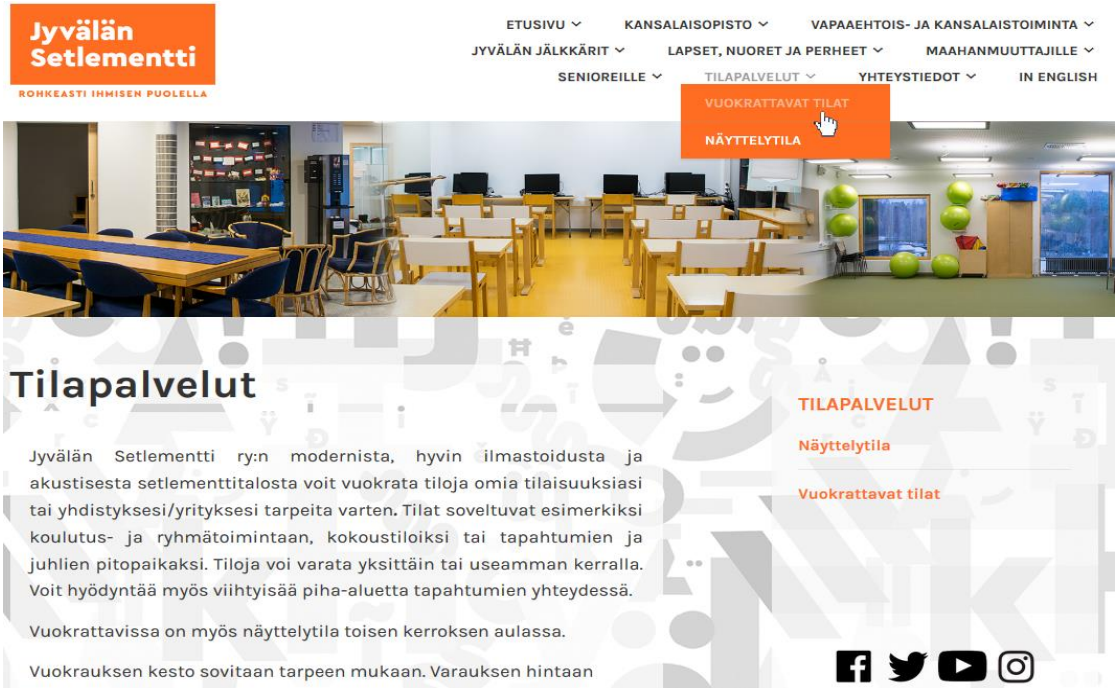
Muiden perussivujen sisältö rakentuu page.php:n kautta. Page.php:ssä itsessään ei ole suuremmin toimintoja, mutta se koostaa sivun header- ja footer-osista, sekä sivupalkista, ja hakee sivun sisällön components/page-kansion content-page.php:sta. Page.php:n toimintaan ei tehty muutoksia, joten sitä ei kopioitu lapsiteemaan, vaan käytössä on isäntäteeman sivupohja.

Näkyvimmät muutokset tulivat sivupalkkiin. Sidebar.php:n kautta sivupalkkiin tulostetaan linkit sivuihin, jotka kuuluvat samaan kategoriaan kuin auki oleva sivu tai kurssivalikko, jos sivu kuuluu kurssit-kategoriaan. Tieto siitä, kuuluuko sivu kurssit-kategoriaan, haetaan functions.php:hen tehdyn onko_kurssit () -funktion kautta, ks kuvio 13. Onko_kurssit-funktiossa haetaan tieto, onko kyseinen sivu nimeltään kurssit käyttäen is_page(’kurssit’) -funktiota ja palautetaan ’true’ jos on. Jos ei ole, niin tutkitaan, kuuluuko artikkeli kurssit-kategorian alle hakemalla artikkelin id:n kautta sen kategoria, ja get_category_parents () -funktiolla haetaan lista kyseisen kategorian yläkategorioista. Tämän jälkeen tutkitaan PHP:n strpos-funktiolla, löytyykö kategorialistasta ’kurssit’. Jos löytyy, eikä avattu sivu ole myöskään arkisto, palautetaan ’true’.

```
function onko_kurssit() {
   global $wp_query;

   if (is_page('kurssit')) {
      return True;
   }

   $post = get_the_ID
   $cat_Id = get_the_category($post);

   if (count($cat_Id)) {
      $categories = get_category_parents( $cat_Id[count($cat_Id)-1], TRUE, ",");

      if ((strpos($categories, 'kurssit') !== FALSE) && !( isset( $wp_query ) && (bool)
$wp_query->is_posts_page )) {
         return True;
      }
   }
```

Kuvio 13. Onko_kurssit -funktio

Lisäksi sivupalkkiin, linkkien ja kurssivalikon alle, tehtiin Jyvälän sosiaalisen median linkit sisältävä plugin-vimpain, joka näkyy kuviossa 14. Perussivut toteutettiin sivuina (page). Kuvion (kuvio x) ylävalikossa näkyvät linkit ovat yläkategorioiden sivuja, niiden alta aukeavat alakategorioihin kuuluvat sivut.



Kuvio 14. Perussivujen yleisilme

5.6.4 Kurssivalikko

Kurssitoiminnot toteutettiin niin, että jokainen kurssi on oma artikkelinsa, joka kuuluu johonkin kategoriaan ja mahdollisesti alakategoriaan. Esimerkiksi rumpuopetuskurssi kuuluu musiikkikategoriaan ja sen alakategoriaan rumpu. Tällöin kursseja voidaan listata kategorioittain. Sidebar.php:ssa kurssit listataan functions.php:n listaakurssit () -funktiolla. Kurssikategoriat haettiin $categories -muuttujaan funktiolla get_categories($args). Kurssit haluttiin listata aakkosjärjestyksessä, joten argumenteissa käytettiin ’orderby’ -parametrin arvoa ’title’. Pääkategoriat listattiin silmukassa, kuten näkyy kuviossa 15, ja haettiin alakategoriat.

```
foreach($categories as $category) {
      echo '<h3 class="kurssiotsikko"><a href="' . get_category_link( $category-
>term_id ) . '" title="' . sprintf( __( "Näytä kaikki kurssit kategoriasta %s" ), $category-
>slug ) . '" ' . '>' . $category->name.'</a></h3><span>';

        $catID2 = get_cat_ID($category->name);

        // Hae kategoriat, parenttina kategoria. Ei näytetä jos ei sisällä kursseja
        $argscat = array('parent' => $catID2,
                  'hide_empty' => True);

        // Yhdistä 'perusargumentit' + alakategoria (argscat)
        $args2 = array_merge($args, $argscat);
        $cate = get_categories($args2);
```

Kuvio 15. Pääkategorioiden listaus ja alakategorioiden argumentit

Kaikki pääkategoriat haluttiin listata, joten ’hide_empty’-parametrin arvoksi asetettiin ’false’. Sen sijaan alakategorioissa käytettiin arvoa ’true’ jolloin vain ne kategoriat, joissa on sisältöä, näytetään. Lisäksi alakategorioissa haluttiin listata kategorian etusivu eri tavalla kuin kurssin sivu niin, että esimerkiksi Rumpuopetus-kurssi näkyy normaalina linkkinä, mutta Rumpu-kategorian linkin eteen piirretään kaksoisnuoli. Tämä toteutettiin niin, että jos artikkelin nimi on sama kuin kategorian nimi, sen eteen lisätään nuoli. Tätä varten nimet piti saada muunnettua pieniksi kirjaimiksi, jonka pitäisi normaalisti onnistua PHP:n mb_strtolower () -funktiolla. Käytännössä artikkelin nimi ei kuitenkaan tällä, eikä muilla testatuilla tavoilla, muuttunut pieniksi kirjaimiksi. Ratkaisuksi löytyi, että artikkelin nimeä hakiessa the_title () -funktiolla on oltava parametreinä $before="", $after="", $echo=false, kuten näkyy kuviossa 16.

```
foreach($cate as $category2) {
        echo('<h3 class="kurssiotsikko"><a href="'.get_category_link( $category2
        ->term_id ).'">'.$category2->name.'</a></h3><span>');

        query_posts('category_name='.$category2->slug);

        while (have_posts()) : the_post();
          $titteli = the_title($before="", $after="", $echo=false);

          // Jos postin nimi sama kuin kategoria -> kategoria-linkki
          if (mb_strtolower($titteli) == mb_strtolower($category2->name)) {

            $category_id = get_cat_ID( $titteli );

            $category_link = get_category_link( $category_id );
            echo'<li>'.$nuoli.'<a  class="kurssivalinta" href="';
            echo $category_link;
            echo'">';
            echo $category2->name.'-kategoria</a></li>';
          } else {
            // Ruutuun kurssisivun linkki
            echo'<li><a  class="kurssivalinta" href="';
            echo the_permalink();
            echo'">';
            echo the_title().'</a></li>';
          }
```

Kuvio 16. Listaakurssit-funktion alakategorioiden listaus

Kurssivalikossa kurssin nimeä klikatessa siihen liittyvä sisältö, eli artikkeli, aukeaa ruutuun. Valikkoon haluttiin aluksi näkyville pelkästään yläkategoriat ja niiden alle klikkaamalla aukeavat alakategoriat, joiden alle kurssit. Toiminto toteutettiin yksinkertaisella jQuery-koodilla, ks kuvio 17, jossa kurssiotsikko -luokkaan kuuluvia kurssikategorioita seuraavat span-tagin sisällä olevat elementit piilotetaan ja otsikkoa klikatessa niiden näkyvyyttä muutetaan.

```
jQuery(document).ready(function(){
  console.log("document ready");

  jQuery("h3.kurssiotsikko").click(function(e){
    e.preventDefault();
    jQuery(this).next("span").toggle("fast");
  });
});
```

Kuvio 17. Kurssivalikon jQuery-koodi

Skripti jyvala-script.js liitettiin functions.php:n add_action(’wp_enqueue_scripts’,’lisaa_jyvala_scripti’) -koukulla ja rekisteröitiin lisaa_jyvala_scripti-funktiossa, ks kuvio 18. Lapsiteeman polku saatiin hakemalla isäntäteeman polku get_template_directory_uri () -funktiolla, ja vaihtamalla teeman nimi ’karuna’ PHP:n str_replace:lla muotoon ’jyvala’.

```
function lisaa_jyvala_scripti() {
   $templatedir = get_template_directory_uri();
   $dir = str_replace('karuna', '', $templatedir).'jyvala';

   wp_register_script('jyvala-script', $dir . '/jyvala-script.js', array('jquery'),'1.0' );
   wp_enqueue_script('jyvala-script');
}
```

Kuvio 18. jQuery-skriptin lisääminen teeman käyttöön

Artikkeleiden sivupohja on single.php. Kurssisisällön yläreunaan haluttiin tulostaa kategoriat, joihin kurssi kuuluu. Esimerkiksi Hyvää oloa ja virkeyttä laulaen -kurssin alkuun haluttiin polku Kurssit->Musiikki->Laulu, ks kuvio 19.

Kuvio 19. Esimerkki kurssin sivusta

Kurssiin liittyvät kategoriat haettiin listaksi wp_get_post_categories () -funktiolla, sen jälkeen muodostettiin silmukassa listasta taulukko, joka sisältää kategorioiden nimet, slugit ja id:t, ja tulostettiin sitten silmukassa kategoriat järjestyksessä, ks. kuvio 20. Slug tarkoittaa WordPressissä artikkelin nimeä tietyn muotoisena. Esimerkiksi Laulu raikaa 5 krt –artikkelin slug olisi 'laulu-raikaa-5-krt'.

```
// Tehdään kategorioista taulukko kurssipolkua varten
foreach($post_categories as $category){
      $categories = get_category( $category );
      $kurssit[] = array( 'name' => $categories->name, 'slug' => $categories->slug, 'id'
=> $categories->cat_ID, );
   }

   // Jos postaus on kurssi, YLÄREUNAAN POLKU: (Kurssit > Musiikki > Laulu)
   if ($kurssit[0]['name'] == 'Kurssit') {

      // Generic-icon -nuoli kategorioiden väliin
      $nuoli = ' <span class="genericon genericon-fastforward" style="color: #FF6D22;
vertical-align: baseline;"></span> ';

      // Tulostetaan kaikki artikkeliin liittyvät kategoriat järjestyksessä
      for ($i = 0; $i < count($kurssit); $i++) {
         $category_link = get_category_link( $kurssit[$i]['id'] );
         echo $nuoli.'<a href="'.$category_link.'">'.$kurssit[$i]['name'].'</a>';
      }
```

Kuvio 20. Kategoriapolku

Aluksi polku ei tulostunut oikein vaan esimerkiksi edellä mainitun kurssin poluksi tulostui Kurssit->Laulu->Musiikki, vaikka hierarkian mukaan Musiikki on Laulun yläpuolella. Tämä johtui siitä, että funktio listaa perusasetuksillaan artikkelit aakkosjärjestyksessä niiden hierarkiasta huolimatta. Hierarkkinen järjestys saadaan käyttämällä parametriä 'orderby' => 'term_id'.

Single.php:ssa artikkelin varsinainen sisältö haetaan tiedostosta components/post/content.php funktiolla get_template_part ().

5.6.5 Kurssi-ilmoittautuminen

Kunkin kurssin kohdalle haluttiin mahdollisuus ilmoittautumiseen. Ilmoittautumistoiminto saatiin osittain automatisoitua niin, että ’Ilmoittaudu kurssille’ -linkki muodostuu kurssin esittelyn alle, jos artikkeliin on liitetty kurssinumero. WordPressin artikkelien muokkaus -toiminnossa on mahdollisuus liittää artikkeliin lisäkenttiä (custom field). Tätä ominaisuutta hyödynnettiin niin, että jos artikkelilla on lisäkenttä nimeltä Kurssinumero ja kentän arvona numero, linkki muodostuu näiden perusteella content.php:ssa. Kurssille on mahdollista asettaa myös lisäkenttä Kurssi_taynna ja sen arvoksi kurssin numero. Tällöin kyseisen kurssin alle linkiksi tulee ’Kurssi täynnä, ilmoittaudu varalle’. Toisinaan samalla kurssilla voi olla useita toteutuksia, joilla on eri numerot, jolloin täynnä olevan kohdalla ilmoitetaan sen olevan täynnä ja jos toisessa toteutuksessa on tilaa, näkyy sen kohdalla vain ’Ilmoittaudu kurssille’. Content.php:ssa haetaan ensin listaan kaikkien Kurssi_taynna -kenttien arvot ja sitten Kurssinumero -kentät. Sen jälkeen listat käydään läpi silmukassa, ja jos kurssinumeroissa oleva arvo löytyy kurssi_taynna -arvoista, sen kohdalle luodaan linkki joka ilmoittaa varalle ilmoittautumisesta ja muussa tapauksessa luodaan linkki ilmoittautumiseen, ks. kuvio 21.

```
// Haetaan kurssi_taynna -lista
 $kurssi_taynna = get_post_meta( get_the_ID(), 'Kurssi_taynna' );

// Haetaan lisäkentistä kurssit
$kurssinrot = get_post_meta( get_the_ID(), 'Kurssinumero' );

// Käydään läpi kurssit, jos numero 'kurssi_taynna' -listassa -> varalle,
// muutoin 'Ilmoittaudu kurssille'
foreach ( $kurssinrot as $key => $value ) {
        if (strpos(implode(",",$kurssi_taynna), $value) !== FALSE) {
           $ilmo = "KURSSI TÄYNNÄ! Ilmoittaudu varalle ";
        } else {
           $ilmo = "Ilmoittaudu kurssille ";
        }

        echo '<a href="https://www.opistopalvelut.fi/
        jyvala/search.php?l=fi&search='.$value.'">'.$ilmo.$value.'</a><br>';
}
```

Kuvio 21. Kurssi-ilmoittautumislinkkien tulostus

5.6.6 Jyvälän some-plugin

Jyvälän Setlementin sivuille toteutettiin myös plugin, joka näyttää Facebook-, Twitter-, YouTube- ja Instagram -kuvakkeet, joista klikkaamalla pääsee katsomaan Jyvälän päivityksiä. Plugin haluttiin sekä footer-osioon, että sivupalkkiin. Footerissa linkkien oli tarkoitus pysyä koko ajan samoina, Jyvälän Setlementti ry:n linkkeinä, mutta tietyillä sivuilla sivupalkissa olevien linkkien haluttiin vaihtuvan riippuen auki olevasta sivusta. Esimerkiksi nuorten sivuille haluttiin linkit nuorten päivityksiin.

Pluginille tehtiin ensin jyvala-feed -kansio WordPressin plugins -kansion alle, ja sinne tiedosto jyvala-feed-plugin.php. Plugin-tiedoston alkuun oli laitettava kommentteihin informaatio-osa, johon tuli pluginin nimi, kuvaus ja tekijä. Plugin oli liitettävä ’widgets_init’ -koukkuun, jonka välityksellä vimpain rekisteröitiin käyttöön, ks. kuvio 22.

```
// Pluginin rekisteröinti
function jyvalafeed_lataa_widget() {
   register_widget( 'jyvalafeed_widget' );
}
// ...liitetään vimpain 'widgets_init'-koukkuun (Action hook)
add_action( 'widgets_init', 'jyvalafeed_lataa_widget' );
```

Kuvio 22. Pluginin rekisteröinti

Pluginin sosiaalisen median linkkien haluttiin olevan vaihdettavissa, joten oli tehtävä myös asetukset -sivu, jonka kautta voi asettaa vakio-osoitteet, mutta myös muutama sivukohtainen osoite. Asetukset-sivua varten jyvala-feed-plugin.php:n luotiin JyvalaSettings-luokka. JyvalaSettings-luokkaan tehtiin jyvalafeed_register_settings_page () -funktio, jolla WordPressin hallintapaneelin sivupaneelin Asetukset -kohtaan saatiin Jyvälä some-plugin -linkki. ks. kuvio 23.

```
public function jyvalafeed_register_settings_page() {
      add_options_page('Jyvälän some-plugin -asetukset', 'Jyvälän some-plugin', 'manage_options', 'jyvalafeed', array( $this, 'jyvalafeed_settings_page' ));
   }
```

Kuvio 23. Pluginin asetukset hallintapaneeliin

Asetukset -sivun käyttäjälle näkyvä puoli koottiin JyvalaSettings-luokan jyvala-feed_settings_page () -funktiossa. Funktioon rakennettiin lomake, josta ylläpitäjä voi vaihtaa peruslinkit, mutta myös määritellä kategorioita, joissa yhdellä tai useammalla linkillä on eri osoite. Jotta arvoja voitiin ladata ja tallentaa, oli ensin tehtävä jyvala-feed_register_settings () -funktio, jossa luotiin ja rekisteröitiin tallennettavat muuttujat. Esimerkiksi yleisen Facebook-osoitteen tietokantaan tallentamista varten luotiin add_option () -funktiolla nimi/arvo -pari ja rekisteröitiin se register_setting () -funktiolla, ks. kuvio 24. Register_setting () -funktiossa määriteltiin arvojen kuuluvan ’jyvalafeed_options_group’ -ryhmään.

```
add_option( 'jyvalafeed_face_osoite', 'oletus');

register_setting( 'jyvalafeed_options_group',  'jyvalafeed_face_osoite',
            array( $this, 'jyvalafeed_callback') );
```

Kuvio 24. Asetukset -sivun tietokantaan tallennettavien tietojen luominen

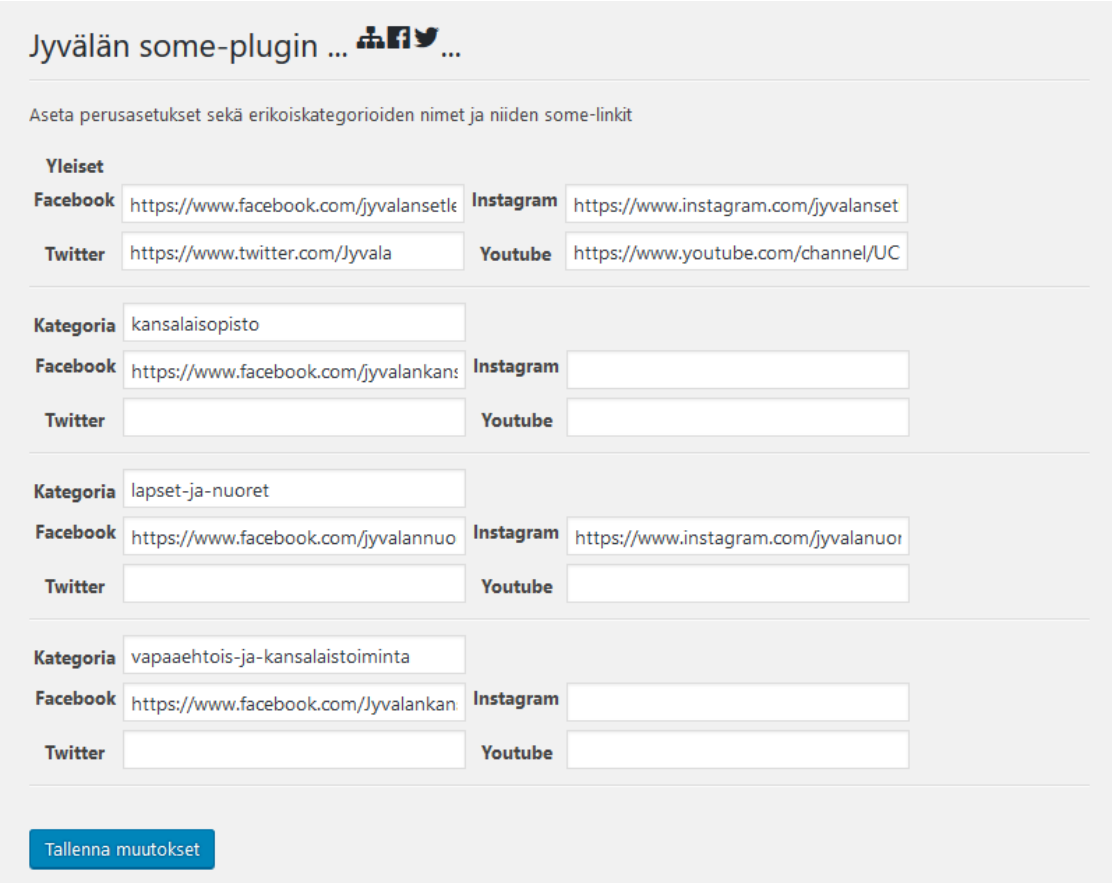
Jyvalafeed_register_settings () -funktio oli liitettävä JyvalaSettings -luokan konstruk-toriin add_action(’admin_init’) -koukkuun. Lisäksi konstruktorissa liitettiin add_action(’admin_menu’) -koukkuun jyvalafeed_register_settings_page -funktio. Register_settings_page () -funktiossa puolestaan asetukset-sivun linkki liitettiin WordPressin hallintapaneelin Asetukset-valikkoon nimellä Jyvälän some-plugin, ks. kuvio 25.

```
public function jyvalafeed_register_settings_page() {
      add_options_page('Jyvälän some-plugin -asetukset', 'Jyvälän some-plugin',
                'manage_options', 'jyvalafeed', array( $this, 'jyvalafeed_settings_page' ));
   }
```

Kuvio 25. Asetukset-sivun liittäminen hallintapaneelin toiminnoksi

Hallintapaneelin asetuksista Jyvälän some-plugin -linkkiä klikatessa aukeava Jyvala-feed_settings_page () -funktio, määrittelee asetukset -sivun ulkoasun. Sivulle luotiin lomake, jossa input-kenttiin voidaan määritellä yleiset Facebook, Instagram, Twitter

ja YouTube -linkit sekä lisäksi kolme kategoriaa, ja niiden linkit, ks. kuvio 26. Jos kategoriakohtainen linkki on tyhjä, käytetään yleistä linkkiä.



Kuvio 26. Jyvälän some-pluginin asetukset-sivu

Lomakkeeseen haetaan tietokannasta aiemmin tallennetut arvot funktiolla settings_fields('jyvalafeed_options_group'). Sen jälkeen input-elementtikohtainen arvo saadaan kenttään komennolla get_option, ks. kuvio 27. Arvojen tallennus tapahtuu yksinkertaisesti WordPressin submit_button () -funktiolla, jota on mahdollista käyttää vain WordPressin ylläpitotoiminnoissa, ei front-endissä. Settings_fields ()- ja submit_button () -funktiot sisältävät tietoturvan kannalta tärkeät nonce-toiminnot, joten niitä ei tarvitse koodiin tässä tapauksessa lisätä (Creating Options Pages n.d.).

```
<label for="jyvalafeed_face_osoite">Facebook</label></th>
 <td>
<input type="text" id="jyvalafeed_face_osoite" size="35" name = "jyvala-
feed_face_osoite" value="<?php echo get_option('jyvalafeed_face_osoite'); ?>" />
```

Kuvio 27. Esimerkki input-kentän arvon asettamisesta

Pluginin asetukset haluttiin myös näkymään suoraan WordPressin hallintapaneelin plugins-luettelossa. Tämä saatiin toteutettua add_filter () -koukun kautta), ks. kuvio 28.

```
// Liitetään pluginin asetukset-linkki plugins-sivulle
add_filter("plugin_action_links_".plugin_basename(__FILE__), 'jyvalafeed_set-
tings_link' );

// Pluginin Asetukset-linkki Plugins-sivulle
function jyvalafeed_settings_link($link) {
  $linkki = '<a href="options-general.php?page=jyvalafeed">Asetukset</a>';
  array_unshift($link, $linkki);
  return $link;
}
```

Kuvio 28. Pluginin asetukset-linkki

Pluginin asetusten luokan lisäksi varsinaiselle pluginille luotiin oma luokka, joka laajentaa WordPressin WP_Widget -luokkaa, komennolla class jyvalafeed_widget extends WP_Widget. Luokan sivulla näkyvän osan eli front-endin toiminnot tulivat widget-nimiseen funktioon. Widget-funktiossa haetaan auki olevan sivun parent-sivut get_post_ancestors -funktiolla $parents-taulukkoon. Sen jälkeen haetaan $parents-taulukon viimeisin tietue ja sen ID. ID:n kautta saadaan haettua parent sivun nimi. Tämän jälkeen haetaan pluginin asetuksista kategorioihin määritellyt arvot ja verrataan niitä nykyisen sivun kategorioihin strpos- ja cat_is_ancestor -funktioilla, ks. kuvio 29. Mikäli auki oleva sivu ei kuulu määriteltyihin kategorioihin, haetaan linkkien perusasetukset. Lopuksi tulostetaan ruutuun sosiaalisen median kuvakkeet ja linkit.

```
// Haetaan sivun "parent"-sivun nimi, jonka mukaan feedien osoitteet
      $post = get_post();
      $parents = get_post_ancestors( $post->ID ); // Artikkelin "parentit"

       // Haetaan ykköstason sivun ID. Id:t lähtee ykkösestä, array nollasta = -1
       $id = ($parents) ? $parents[count($parents)-1]: $post->ID;
       $parent = get_post( $id ); // Haetaan parentin "slug"
       $nimi = $parent->post_name; // Parent-sivun nimi

        // Haetaan kategoriat pluginin asetuksista
        $kat1 = mb_strtolower(get_option('jyvalafeed_kat1'));
        $kat2 = mb_strtolower(get_option('jyvalafeed_kat2'));
        $kat3 = mb_strtolower(get_option('jyvalafeed_kat3'));

// Mikäli ollaan määritellyssä kategoriassa, otetaan sen osoite käyttöön (jos ase-
tettu). Eli jos sivun parentin nimi, tai artikkelin "esi-isä" kuuluu kategoriaan
if ((strpos($nimi, $kat1) !== FALSE) or (cat_is_ancestor_of(0, $kat1) or is_cate-
gory(0))) {

// Jos jyvalafeed_face_osoite1 asetetettu, haetaan tietokannasta
if (get_option('jyvalafeed_face_osoite1'))
           $faceosoite = get_option('jyvalafeed_face_osoite1');
```

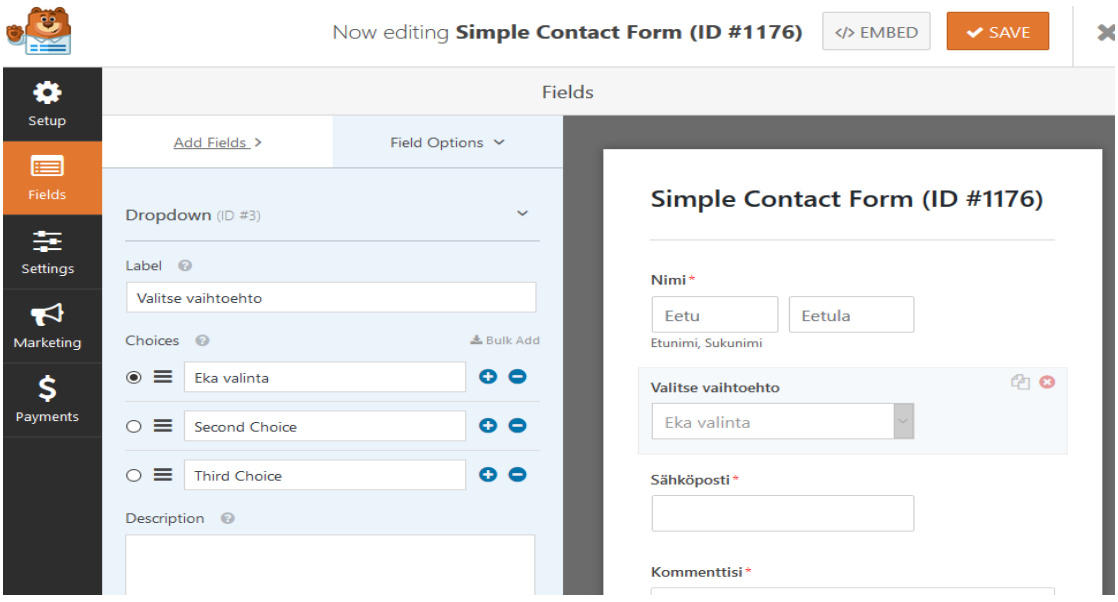
Kuvio 29. Auki olevan sivun kategorioiden hakeminen

Some-kuvakkeet haluttiin näkyviin sekä sivun footeriin että sivupalkkiin, joten plugin sijoitettiin molempiin, kuitenkin sillä erotuksella, että sivupalkin linkkien piti vaihtua tietyissä kategorioissa. WordPressissä ei ole olemassa funktiota, jolla plugin saisi tiedon omasta sijainnistaan, joten linkkien muuttuminen sivupalkin pluginiin piti tehdä toisella tapaa. Tämä toteutettiin niin, että functions.php:hen tehtiin funktiot set_widgetin_sijainti($widget_sijainti) ja get_widgetin_sijainti (), joiden kautta saatiin tieto pluginin sijainnista välitettyä. Sivun footeria ladatessa teema hakee footeriin mahdollisesti asetetut vimpaimet components/footer/footer-widgets.php:sta, joten footer-widget.php:hen lisättiin rivi set_widgetin_sijainti(”footer”) ennen widgetien lataamista. Jyvala-feed-plugin.php:n widget-funktio hakee functions.php:sta tiedon sijainnista rivillä $widget_sijainti = get_widgetin_sijainti(), ja mikäli $widget_sijainti-muuttujan arvo on ”footer”, kategorioiden mukaan määriteltyjä linkkejä ei ladata vaan yleiset linkit. Toisin sanoen sivupalkin pluginia ladatessa sijaintia ei ole määritelty, mutta se määritellään arvoon ”footer” juuri ennen kuin footerin plugin ladataan.

5.6.7 Lomakkeet

Jyvälän sivuilla tarvitaan useita erityyppisiä lomakkeita. Näitä ovat muun muassa palaute-, uutiskirjeen tilaus-, toivo kurssi- ja leiri-ilmoittautumislomake. WordPressiin on saatavilla suuri määrä lomakeplugineita, mutta sellaisen voi toki tehdä itsekin. Jyvälän tapauksessa, kun tarvittavia lomakkeita on useanlaisia ja tulevaisuudessa tarvitaan mahdollisesti lisääkin erilaisille leiri-, kurssi- ym. sivuille, oli järkevää käyttää valmista pluginia, jota pystyisi muokkaamaan tarpeen mukaan.

WordPressin suosituin lomakeplugin on Contact Form 7 yli viidellä miljoonalla latauksellaan. Contact Form 7:n käyttäminen vaatii kuitenkin jossain määrin HTML-tuntemusta. Esimerkiksi yksinkertaisen tekstikentän luominen tapahtuu tyyliin Nimesi:
[text* your-name] . Yksi suosituimpia Contact Form 7:n jälkeen on WPForms, jossa lomakkeiden luominen tapahtuu drag&drop -tyylisesti, ks. kuvio 30. Kun lomake on luotu, se saadaan käyttöön sivulle tai artikkeliin, lisäämällä tägi [wpforms id="*lomakkeen_id_nro*"]. Pluginin ilmaisversiossa, WPForms Litessä on käytettävissä mukavasti erilaisia toimintoja, ja mobiiliresponsiivisuuskin on otettu hyvin huomioon, vaikkakin maksullisella versiolla niitä saa käyttöön vielä enemmän. WPForms Lite on näppärä plugin omien lomakkeiden tekemiseen ja se valittiin käytettäväksi Jyvälän sivuilla. Plugin on tietysti mahdollista vaihtaa myöhemmin, jos siihen koetaan tarvetta.



Kuvio 30. WPForms-pluginin lomakkeen muotoilu

5.7 Hellewi-järjestelmän yhteensopivuus

Hellewin kurssitiedoista on mahdollista saada JSON-muotoista dataa, joka olisi suuresti hyödyksi tietojen käsittelyssä. Valitettavasti JSON-rajapinnan ylläpidosta veloitetaan erikseen, eikä tähän nähty tarpeelliseksi käyttää lisäresursseja. Kursseille ilmoittautumista pystyttiin kuitenkin kehittämään paremmaksi, niin että kunkin kurssin alla olevasta Ilmoittaudu kurssille -linkistä aukeaa kyseisen kurssin sivu Hellewissä. Käytännössä linkki lähettää Hellewin hakutoimintoon kyseisen kurssinumeron. Käyttäjän tarvitsee siis klikata vain Ilmoittaudu kurssille -linkkiä ja seuraavaksi aukeavalta Hellewi-järjestelmän sivulta Ilmoittaudu -linkkiä.

5.8 Toimintojen ja ilmeen yhdenvertaistaminen (Oiva ja Ilona)

Työn edetessä tuli tieto, että Oiva ja Ilona -sivustoa ei sittenkään liitetä uuteen sisällönhallintaan, joten tämä osio jäi pois.

5.9 Mobiilikäytettävyys

Karuna-teemassa, josta Jyvälän teema polveutuu, on itsessään hyvin toteutettu mobiiliresponsiivisuus. Muutoksia tehtiin ainoastaan värimaailmaan ja header-kuvien toimintaan style.css-tyylisivun kautta. Karuna-teema skaalaa kuvat sitä pienemmäksi, mitä pienempi ruutu on. Jyvälän sivuja testatessa tultiin tulokseen, että mobiilikäytössä kuvat ovat liian pieniä. Tätä varten css-tyyliä muokattiin niin, että kuvat eivät enää pienene 768-pikselin koosta alaspäin, ks. kuvio 31. Mobiilikäytettävyyttä testattiin pienellä selainruudulla sekä useammalla puhelimella toimivuuden varmistamiseksi.

```
@media screen and (max-width: 768px){
   .attachment-karuna-hero {
      min-width: 768px;
      height: auto;
      overflow: hidden;
   }
   img.custom-header {
      min-width: 768px;
      height: auto;
      overflow: hidden;
   }
}
```

Kuvio 31. Hero- ja header-kuvien koon määrittely CSS-tyylisivulla

5.10 Uuden sivuston käyttöönotto

Ennen sivuston siirtämistä Jyvälän palvelimelle teemaa ja osittain sisältöäkin rakennettiin virtuaalikoneeseen asennettuun WordPressiin. Sivuston siirtäminen palvelimelle täytyi tehdä vaiheittain ja tietyssä järjestyksessä, jotta se onnistui ongelmitta. Ensimmäisenä asennettiin WordPress, ja ladattiin ja asennettiin siihen Karuna-teema WordPressin ohjauspaneelin kautta. Seuraavaksi voitiin asentaa Jyvälä-teema kopioimalla jyvala-kansio themes-kansion alle ja otettiin se käyttöön asetuksista. Tämän jälkeen virtuaalikoneen WordPress-tietokannasta otettiin kopio MySQL Workbench -ohjelmalla ja muutettiin tietokannan ’siteurl’- ja ’home’- tietueiden arvot vastaamaan sivuston todellista osoitetta. Tämän jälkeen tietokannan kopio siirrettiin palvelimen SQL-tietokantaan phpMyAdmin-ohjelmaa käyttäen. Lopuksi oli vielä vaihdettava wp-config.php:hen tietokannan etuliite kopioidun tietokannan etuliitteen mukaiseksi.

6. Tulosten tarkastelu

6.1 Saavutetut tavoitteet

Ensisijainen tavoitteeni oli saada toteutettua toimeksiantajalle odotuksien mukainen verkkosivusto, johon voin itsekin olla tyytyväinen. Sivuston toteutus onnistuikin odotusten mukaisesti. Yleisilmeestä saatiin moderni ja samalla käyttäjälle huomattavasti aiempaa selkeämpi. Päivittämisen kannalta rakenteesta tuli myös suoraviivaisempi. Myös Karuna-teeman mobiilikäytettävyys testattiin, ja todettiin pienten muutosten myötä hyvin toimivaksi. Työn aikana sivuston vaatimusmäärittely muuttui jonkin verran, mutta muutokset eivät olleet perustavanlaatuisia. Välillä luonnollisesti nousi myös uusia ideoita, mitä olisi hyvä olla tai miten jokin voisi toimia vielä paremmin, ja näitä pyrittiin toteuttamaan. Sivustolle haluttiin myös useanlaisia lomakkeita. Tässä päädyttiin käyttämään valmista WPForms Lite -pluginia, joka osoittautui sopivan monipuoliseksi ja helppokäyttöiseksi. Pluginin kautta pystytään luomaan haluttuihin kohtiin juuri sellaisia lomakkeita kuin sivustolla tarvitaankin.

Toimeksiantaja oli tyytyväinen lopputulokseen ja voidaankin todeta, että sisällönhallinnan ja juuri WordPressin käyttäminen alustana oli toimiva ratkaisu. Halutut toiminnallisuudet saatiin tehtyä tavoitteista tinkimättä ja palaute työstä oli positiivista. Ulkoasua pidettiin onnistuneena ja kurssivalikko koettiin toimivaksi, samoin sosiaalisen median plugin asetuksineen. Myös sisällön tuottaminen uudelle sivustolle saatiin jo hyvään alkuun Jyvälän Setlementissä, mikä tukee tehtyä päätelmää WordPressin matalasta käyttöönottokynnyksestä. Myös kurssi-ilmoittautumista saatiin parannettua entisestä, mikä oli yksi tavoitteista. Aiemmin ilmoittautumislinkki vei Hellewi-järjestelmään, josta kurssi piti etsiä uudestaan, nyt linkistä päästään suoraan oikean kurssin kohdalle. Työ antoi hyvän poikkileikkauksen sisällönhallintajärjestelmistä, erityisesti WordPressistä, sekä PHP-kielestä. Myös työskentely Jyvälän Setlementin viestintätiimin kanssa oli hyvä kokemus toimivasta tiimityöstä. Verkkosivuston toivon toimivan myös hyvänä referenssinä tulevaisuuden työtehtäviä ajatellen.

6.2 Ohjelmoinnin haasteet

Henkilökohtainen tavoitteeni oli myös tutustua syvällisemmin eri tekniikoihin. Työtä tehdessä jouduinkin oppimaan vastaan tulleiden haasteiden kautta lisää WordPressin ominaisuuksista, funktioista ja ohjelmointitavoista, kuten myös yleisesti PHP-kielestä. Yksi haastava osa-alue oli selvittää mihin tiedostoon mikäkin muutos on tehtävä, koska sivut koostuvat monesta osasta ja toimivat omanlaisellaan rakenteella. Esimerkiksi page.php:ssa on get_template_part(’components/page/content/’ , ’page’), joka kertoo että sivun osa haetaan components/page -hakemistopolusta ja tiedoston nimi on content-page.php. Oikeaa tiedostoa joutui välillä jäljittämään seuraamalla näitä funktioita. Sivupohjien hierarkiasta löytyy yleisiä kuvauksia WordPressin sivuilta, mutta nekin vaihtelevat teemasta riippuen.

Välillä vastaan tuli myös yllättäviä ongelmia, joiden ratkaisu oli hyvin pienestä kiinni. Esimerkiksi pluginin ohjelmoinnissa, kun pluginia yritettiin ottaa käyttöön WordPressin hallintapaneelista, tuli virheilmoitus ”Warning: Cannot modify header information - headers already sent”. Tämä johtui siitä, että plugin-tiedoston loppuun oli laitettu sulkeva PHP-tagi ”?>”. Tätä lopputagia ei pluginissa saa olla, vaikka tiedoston alussa PHP:n aloittava ”<?php” onkin oltava. Toisella kertaa sama virheilmoitus tuli pelkästään siitä syystä, että tiedoston alussa oli ylimääräinen tyhjä rivi.

6.3 Jatkokehitys

Sivuston jatkokehitystä ajatellen kursseihin liittyviä toimintoja voisi kehittää pidemmälle, jos Hellewi-järjestelmän JSON API voitaisiin ottaa käyttöön. Kurssien sisältö on edelleen valitettavasti laitettava erikseen sekä Hellewiin, että WordPressin tietokantaan. JSON API:lla kurssien sivuja voitaisiin automatisoida niin, että sisältö tarvitsisi kirjoittaa vain kertaalleen. Toinen kehityskohde liittyy valikkorakenteeseen. Tällä hetkellä etusivua ja kurssisivuja lukuun ottamatta sivupalkkiin listautuu auki olevan sivun kanssa samaan parent-kategoriaan kuuluvat sivut. Pääosin tämä on hyvä käytäntö. Esimerkiksi Vapaaehtois- ja kansalaistoiminta -sivun ollessa auki, sen kategoriaan kuuluvat linkit näkyvät sivupalkissa, muiden muassa Mummola, jonka parent-sivu kyseinen Vapaaehtois- ja kansalaistoiminta on. Kehityskohde tulee kuitenkin siinä, että

Mummola-linkki löytyy myös Lapset, nuoret ja perheet -valikossa, ja olisi hyvä, jos se näkyisi myös sivupalkissa, kun Lapset, nuoret ja perheet -sivu avataan. Se ei kuitenkaan näy, koska kyseinen sivu ei ole Mummolan parent, eikä sivulla voi olla montaa parent-sivua. Yksi ratkaisuvaihtoehto voisi olla sivupalkin sisällön hakeminen valikkorakenteen mukaan niin, että jos esimerkiksi Mummola-sivu avataan Lapset, nuoret ja perheet -valikon kautta, listattaisiin sivupalkkiin kyseisen valikon linkit. Toinen mahdollinen tapa voisi olla lisätä Mummola-sivun lisäkenttiin ne sivut, joissa linkki halutaan esille.

Lähteet

About CMS Made Simple. N.d. CMS Made Simple –sisällönhallintajärjestelmän esittelysivu. Viitattu 30.4.2018. Http://www.cmsmadesimple.org/about/cms-made-simple/

Announcing CMSMS 1.11.4 Fernandina. 2012. CMS Made Simple –sivuston tiedote. Viitattu 27.10.2016. Http://www.cmsmadesimple.org/2012/12/Announcing-CMSMS-1-11-4-Fernandina/

Ardourel, F. 2016. How to choose a CMS: a simple guide. CMS Critic –sivuston artikkeli. Viitattu 1.11.2017. Https://www.cmscritic.com/how-to-choose-a-cms-simple-guide/

Child Themes n.d. WordPress.orgin Codex-sivusto. Viitattu 8.4.2018. Https://codex.wordpress.org/Child_Themes

CMS Made Simple 2.x Official Documentation, N.d. CMS Made Simple -ohjesivusto. Viitattu 30.4.2018. Https://docs.cmsmadesimple.org/

CMS Made Simple Requirements, N.d. CMS Made Simple -vaatimussivusto. Viitattu 18.2.2018. Https://docs.cmsmadesimple.org/installation/requirements

Couch CMS. N.d. Couch CMS sisällönhallintajärjestelmän kotisivu. Viitattu 21.1.2018. Https://www.couchcms.com/

Couch CMS Requirements, N.d. Couch CMS -sisällönhallintajärjestelmän ohjesivusto. Viitattu 18.2.2018. Http://docs.couchcms.com/requirements.html

Creating Options Pages. N.d. WordPress.orgin Codex-sivusto. Viitattu 13.5.2018. Https://codex.wordpress.org/Creating_Options_Pages

Doyle, M. 2010. Beginning PHP 5.3. Indianapolis: Wiley Publishing, Inc.

Drupal 8 system requirements. N.d. Drupal-sisällönhallintajärjestelmän ohjesivusto. Viitattu 30.4.2018. Https://www.drupal.org/docs/8/system-requirements/drupal-8-php-requirements

Drupal-julkaisujärjestelmä. N.d. KWD Digital Oy:n verkkosivu. Viitattu 14.2.2018. Https://www.kwd.fi/palvelut/tekninen-toteutus/drupal

Drupal vs WordPress CMS Comparison Which To Choose? 2017. Kasa Reviews -sivuston artikkeli. Viitattu 11.2.2018. Https://www.kasareviews.com/drupal-vs-wordpress-comparison/Hellewi. N.d. Hellewi-kurssienhallintajärjestelmän sivusto. Viitattu 24.3.2018. Http://hellewi.fi

Heijmans, M. 2018. Blogi Yoast.comin sivuilla. Viitattu 12.5.2018. Https://yoast.com/wordpress-security/#goodpasswords

Historia. N.d. Historia-sivu suomen setlementtiliiton verkkosivustolla. Viitattu 9.10.2017. Https://www.setlementti.fi/setlementtiliitto/setlementtiliitto/historia

Installing Joomla, N.d. Joomla-dokumentaatiosivu. Viitattu 18.2.2018. Https://docs.joomla.org/J3.x:Installing_Joomla

Introducing CMSMS 2.0. N.d. . CMS Made Simple –sivuston tietosivu. Viitattu 21.1.2018. Https://docs.cmsmadesimple.org/introducing-cmsms-2-0

Joomla. N.d. Joomla-sisällönhallintajärjestelmän suomalainen verkkosivusto. Viitattu 21.2.2018. Https://www.joomla.fi

Joomla! Documentation. N.d. Joomla-sisällönhallintajärjestelmän ohjesivusto. Viitattu 8.4.2018. Https://docs.joomla.org/PHP_essentials

Jyvälän Setlementti. N.d. Jyvälän Setlementti ry:n kotisivu. Viitattu 6.10.2017. Http://www.jyvala.fi

Kallio, E. 2013. Verkkopalveluyritys Sofokuksen blogi. Viitattu 5.4.2018. Https://www.sofokus.com/fi/blogi/django-ei-ole-sisallonhallintajarjestelma

Mening, R. 2015. WordPress vs Joomla vs Drupal. CMS-järjestelmien vertailu web-sitestup.org-sivustolla. Viitattu 18.2.2018. Https://websitesetup.org/cms-comparison-wordpress-vs-joomla-drupal/

Miksi me käytämme Drupalia. N.d. Kukumo Creative -mainostoimiston verkkosivu. Viitattu 11.2.2018. Http://www.kukumo.fi/ajatuksia/miksi-me-kaytamme-drupalia

Niemi, T. 2015. Verkkopalveluyritys Sofokuksen blogi. Viitattu 5.4.2018. Https://www.sofokus.com/fi/blogi/sovelluskehys-verkkoraatalin-tyokalupakki

Oiva ja Ilona. N.d. Jyvälän Oiva ja Ilona -sivusto. Viitattu 9.10.2017. Http://www.oiva-ilona.fi

Petreski, P. 2015. WordPressin nonce-funktiota käsittelevä artikkeli tipsandtricks-hq.com -sivustolla. Viitattu 13.5.2018. Https://www.tipsandtricks-hq.com/introduction-to-wordpress-nonces-5357

Poranen, A. 2015. Ohjelmointi WordPress-ympäristössä. Opinnäytetyö. Mikkelin ammattikorkeakoulu, tietojenkäsittelyn koulutusohjelma. Viitattu 8.4.2015. Https://www.theseus.fi/bitstream/handle/10024/87802/Poranen_Aapeli.pdf

Pouru, M. 2012. Web-sisällönhallintajärjestelmän käyttöönotto. Opinnäytetyö. Satakunnan ammattikorkeakoulu, tietojenkäsittelyn koulutusohjelma. Viitattu 3.4.2018. Http://www.theseus.fi/bitstream/handle/10024/40507/pouru_maarit.pdf

Requirements, N.d. WordPress.org:in ohjesivusto. Viitattu 18.2.2018. Https://wordpress.org/about/requirements/

Schäferhoff, N. 2015. 75+ Brands, Celebrities, And Famous Websites Using WordPress. Artikkeli Torque Magazine -sivustolla. Viitattu 18.2.2018. Https://torquemag.io/2015/08/brands-celebrities-famous-websites-using-wordpress/

System requirements. N.d. Drupal.orgin dokumentaatio. Viitattu 18.2.2018. Https://www.drupal.org/docs/8/system-requirements

Tolvanen, P. 2007. Web-sisällönhallintajärjestelmä – ominaisuudet ja käyttöönotto. Pro gradu -tutkielma. Jyväskylän yliopisto, tietojärjestelmätiede. Viitattu 2.4.2018. Http://www.projekti55.fi/web-sisallonhallintajarjestelma/

Tolppi, T. 2013. Www-sivuston rakentaminen Drupal-sisällönhallintajärjestelmällä. Kemi-Tornion ammattikorkeakoulu, tietojenkäsittelyn koulutusohjelma. Viitattu 12.2.2018. Http://urn.fi/URN:NBN:fi:amk-201305158745

Upgrading old CMS Made Simple versions. N.d. CMS Made Simple -julkaisujärjestelmän kotisivut, upgrading-osio. Viitattu 21.10.2017. Https://docs.cmsmadesimple.org/upgrading/old-versions.

Usage of content management systems for websites. N.d. Tilastoja tarjoavan W3Techs-sivuston sisällönhallintajärjestelmien vertailu. Viitattu 30.4.2018. Https://w3techs.com/technologies/overview/content_management/all

Wappalyzer. N.d. Wappalyzer tilastosivuston CMS-sivu. Viitattu 30.4.2018. Https://www.wappalyzer.com/categories/cms

What is a WordPress Child Theme? Pros, Cons and More. 2013. WPbeginner-blogisivusto. Viitattu 8.4.18. Http://www.wpbeginner.com/beginners-guide/wordpress-child-theme-pros-cons

Writing a Plugin, N.d. Wordpress.orgin Codex-sivusto. Viitattu 12.5.2018. Https://codex.wordpress.org/Writing_a_Plugin